BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF INTELLIGENT SYSTEMS
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

CONTROL OF POSITIONABLE PLATFORM FOR EYE CENTERING IN IMAGE

ŘÍZENÍ POLOHOVATELNÉ PLATFORMY PRO VYSTŘEDĚNÍ OKA V OBRAZU

MASTER'S THESIS
DIPLOMOVÁ PRÁCE

AUTHOR **Bc. PATRIK MAGDOLEN**
AUTOR PRÁCE

SUPERVISOR **Prof. Ing. MARTIN DRAHANSKÝ, Ph.D.**
VEDOUCÍ PRÁCE

BRNO 2018

Zadání diplomové práce/21441/2017/xmagdo01

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav inteligentních systémů Akademický rok 2017/2018

Zadání diplomové práce

Řešitel: **Magdolen Patrik, Bc.**

Obor: Inteligentní systémy

Téma: **Řízení polohovatelné platformy pro vystředění oka v obrazu**
Control of Positionable Platform for Eye Centering in Image

Kategorie: Umělá inteligence

Pokyny:

1. Prostudujte literaturu týkající se detekce kruhových útvarů (oka) ve videu a ovládání motorů 3D polohovatelné platformy.
2. Navrhněte algoritmický postup pro detekci oka ve videu a výpočet jeho pozice s následným řízením 3D polohovatelné platformy, aby se sjednotily optické osy oka a kamerového systému.
3. Navržený postup z předchozího bodu implementujte, včetně propojení na 3D polohovatelnou platformu přes API.
4. Proveďte otestování na několika dobrovolnících a shrňte dosažené výsledky.

Literatura:

- Kim H. et al. *Eye Detection in a Facial Image Under Pose Variation Based on Multi-scale Iris Shape Feature*. Image and Vision Computing, 2017, Vol. 57, pp. 147-164, ISSN 0262-8856.
- Kaur E.H., Lcet P., Sohal J.S. *Iris Localization Using Hybrid Algorithm Containing Circular Hough Transform, Fuzzy Clustering Method and Canny Edge Detector*. International Journal of Advanced Research in Computer Science, 2017, ISSN 0976-5697.

Při obhajobě semestrální části projektu je požadováno:

- Bez požadavků.

Podrobné závazné pokyny pro vypracování diplomové práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Vedoucí: **Drahanský Martin, prof. Ing., Dipl.-Ing., Ph.D.**, UITS FIT VUT

Konzultant: Hájek Josef, Ing., Ph.D., UITS FIT VUT

Datum zadání: 12. července 2018

Datum odevzdání: 31. července 2018

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav inteligentních systémů
612 66 Brno, Božetěchova 2

L.S.

doc. Dr. Ing. Petr Hanáček
vedoucí ústavu

Abstract

Ophthalmology is a branch of medicine that deals with the anatomy, physiology and diseases of the eyeball and orbit. An ophthalmic device for the acquirement and recognition of a human eye characteristics was created by researchers from Faculty of Information Technology. This device can be used either for biometric purposes or for medical purposes as a support diagnostic device. To achieve proper functionality, device must be able to adjust platform position in order to align optic camera with patient's eye. The main focus of this thesis is to design and implement an algorithm for eye centre localisation based on images of the patient's face. The first part of this thesis describes general methods for eye localisation and proposed solution. To achieve requested accuracy, combination of multiple methods is used with adjusted parameters based on platform's features. The second part describes implementation of proposed solution as well as platform control. Multiple databases were used for training and testing of the algorithm. The third part summarises performed experiments. The proposed algorithm was implemented in the C++ language, using OpenCV library. Accuracy and speed of proposed algorithm are suitable for developed platform. In the end, the results are discussed and further improvements are proposed.

Abstrakt

Oftalmologie je odvětví medicíny, která se zabývá anatomií, fyziologií a onemocněními oční bulvy a orbity. Oftalmologický nástroj pro získávání a rozpoznávání lidských očních rysů vytvořili výzkumní pracovníci z Fakulty informačních technologií. Toto zařízení lze použít jako biometrické účely nebo pro lékařské účely jako podpůrné diagnostické zařízení. Pro dosažení správné funkčnosti musí být zařízení schopno nastavit polohu plošiny tak, aby se kamera vyrovnala s očima pacienta. Hlavním zaměřením této práce je navrhnout a implementovat algoritmus pro centrování oka založený na obrazech pacienta. První část práce popisuje obecné metody lokalizace oka a navrhuje se řešení. Pro dosažení požadované přesnosti je použita kombinace více metod s upravenými parametry podle vlastností platformy. Druhá část popisuje implementaci navrhovaného řešení a ovládaní platformy. Pro učení a testování algoritmu bylo použito více databází. Třetí část shrnuje výsledky experimentů. Navrhovaný algoritmus byl implementován v jazyce C++ pomocí knihovny OpenCV. Přesnost a rychlost navrhovaného algoritmu jsou vhodné pro vyvíjenou platformu. Nakonec se diskutuje o výsledcích a navrhuje se další zlepšení.

Keywords

eye localization, eye center localization, biometrics, pupil localization, control of positionable platform

Klíčová slova

lokalizace oka, lokalizace středu oka, biometrie, lokalizace pupily, řízení polohovatelné platformy

Reference

MAGDOLEN, Patrik. *Control of Positionable Platform for Eye Centering in Image.* Brno, 2018. Master's thesis. Brno University of Technology, Faculty of Information Technology. Supervisor Prof. Ing. Martin Drahanský, Ph.D.

Rozšířený abstrakt

Význam biometrie ako metódy autentifikácie a identifikácie, rástol od konca 20. storočia a stal sa de facto štandardom v každodennom živote v čase, kedy boli prvé mobilné telefóny so snímačom odtlačkov prstov uvedené na trh v roku 2013. Ďalšie silné a spoľahlivé metódy identifikácie osôb zahŕňajú rozpoznávanie dúhovky a skenovanie sietnice. Sú používané v mnohých komerčných sektoroch k ochrane proti neoprávnenému prístupu osôb s neuveriteľne vysokou presnosťou.

Zo všetkých možných postihnutí má ztrata zraku najväčší vplyv na každodenný život človeka. Tento dopad je väčší ako ztrata pamäte, hlasu alebo sluchu. Dôvodom je, že 80% zo všetkých senzorických informácií, ktoré ľudský mozog príjma, prichádza cez videnie. Takto se oko stáva najdôležitejším senzorickým orgánom. Skutočnosťou však je, že väčšine prípadov straty zraku možno predchádzať včasným zistením ochorenia a jeho liečbou. Návšteva lekára môže byť pre niektorých ľudí nepríjemná, nielen kvôli strachu z vyšetrenia, ale taktiež kvôli časovej náročnosti. To môže byť aspoň čiastočne odstránané vývojom automatizovaných zariadení, ktoré by mohli takmer bez zásahu lekára zhodnotiť stav pacienta. Tieto zariadenia môžu skenovať ľudské oko a jeho časti alebo dokonca diagnostikovať stav pacienta.

Pre účely rozpoznávania dúhovky a skenovania sietnice je väčšinou používaný špecializovaný hardware s takmer infračerveným osvetlením, ktorý je schopný získať presné umiestnenie dúhovky. Tieto riešenia sú často masívne a drahé. Vďaka neustálemu pokroku vo vývoji, predovšetkým hardwaru, je možné tieto systémy implementovať na nešpecializovaných zaradeniach, akým je napríklad obyčajný stolný počítač spolu s webovou kamerou. Táto práca sa zaoberala témou lokalizácie očného centra. Takýto systém môže byť použitý v oftalmologickém zariadení, ktoré vytvorili výzkumní pracovníci z Fakulty informačných technológií VUT v Brne. Očné zaradenie pre automatické zachytenie očnej sietnice a dúhovky má dva možné spôsoby využitia - buď pre biometrické účely alebo pre lekárské účely ako podporné diagnostické zariadenie. Skladá sa z osvetľovacích zariadení, adaptívneho optického systému a jednotky snímania umiestnených na polohovatelnej plošine. Pre svoju činnosť však vyžaduje vyrovnanie osi oka a snímacej kamery. Toto skenovacie zariadenie spolu s bežne používanými metódami skenovania oka sú popísané v kapitole 2.

Táto práca sa zaoberá lokalizáciou očného centra za účelom vycentrovania osi oka a kamery. Behom posledných desaťročí bola lokalizácia a sledovanie očí jednou z najaktívnejších oblastí výskumu v oblasti počítačového videnia, najmä vďaka jej širokej škále aplikácii. Vzniklo tak aj množstvo metód a prístupov, k riešeniu problému lokalizácie očného centra. Všeobecne tieto metódy môžu byť široko zaradené do dvoch základných kategórií. Prvou sú aktívne lokalizačné systémy, ktoré typicky vyžadujú podporné zariadenie umiestnené na hlave alebo infračervenú kameru. Druhou sú pasívne lokalizačné systémy, ktoré sa usilujú o lokalizáciu oka založenej iba na obrázkoch dodávaných z video kamery. To sa stalo možným z dôvodu zvýšenej rýchlosti počítačových procesorov, zdokonalených techník počítačového videnia a dostupnosti vysokovýkonných digitálnych kamier. Podľa metódy lokalizácie oka ďalej rozdeľujeme tieto metódy na metódy založené na črtoch, ktoré využívajú príslušné očné vlastnosti, akými sú okraje, rohy očí alebo body, ktoré sú vybrané na základe špecifických odoziev filtrov. Ďalej sú to metódy založené na modeloch, ktoré využívajú predošlý model očného holistického vzhľadu a okolitých štruktúr, dokonca aj tvár. Poslednou katógiou sú hybridné metódy, ktoré kombinujú predošlé prístupy. Všeobecné prístupy k lokalizácií oka so základnými algoritmami sú opísané v kapitole 3.

Ukázalo sa, že existuje množstvo algoritmov zaoberajúcich sa lokalizáciou oka, ktoré poskytujú presné a spoľahlivé výsledky. Avšak iba málo z nich sa snaží nájsť presný stred

očnej zrenice. Z tohto dôvodu bolo v tejto práci nutné použiť kombináciu dvoch algoritmov a dosiahnuť tak vyžadovanú presnosť. Prvý implementovaný algoritmus je Timm & Barth, ktorého výhodami sú spoľahlivosť a súčastne má relatívne nízke požiadavky na výpočetný výkon. Pre svoju činnosť využíva gradient, z ktorého počíta a ohodnocuje kandidátne očné centrá. Nakoniec, ako očné centrum vyberie najlepšie ohodnotený bod. Jeho hlavnou úlohou teda je nájdenie oka a približnú polohu zrenice a následne sa aplikuje druhá metóda, ktorá v označenom regióne ustanoví pozíciu úplného stredu zrenice. Metóda, ktorá bola pre tento účel vybraná je Terence Brounsov algoritmus, ktorý funguje iba s vhodne pripravenými obrázkami, a preto pracuje výborne ako pomocný, podporný algoritmus. Svoju presnosť dosahuje tak, že v obraze rozpoznáva charakteristiky oka, ktoré následne klasifikuje medzi kontúry zrenice a oka. Po ich určení sa na okraje zrenice prispôsobí elipsa, ktorej stred je stredom zrenice oka. Implementované riešenie taktiež opravuje chyby v obrázkoch oka, ktoré vznikli ako odrazy silných zdrojov svetla pri snímaní. Uvedená kombinácia algoritmov dosahuje veľmi pozitívne výsledky, zlyhávajúc iba v zložitých a neprehľadných obrázkoch. Po úspešnej lokalizácií nasleduje výpočet pozície a odoslanie potrebných príkazov pre platformu. Implementované riešenie teda ukazuje, že hľadanie centra oka a úprava polohy platformy sa dá vyriešiť bežným stolným počítačom spoločne s kamerou. Riešenie bolo implementované v programovacom jazyku C++ s využitím knižnice OpenCV pre spracovanie a manipuláciu s obrázkami. Detaily na implementáciu vybraných metód sú popísané v kapitole 4.

Implementovaný systém je potom vyhodnotený najprv pomocou dátových sad, následne pomocou testovania s dobrovoľníkmi na platforme. Kapitola 5 obsahuje zhrnutie výsledkov a ich analýzu. Na testovanie implementovaného algoritmu boli použité dve databázy. Rozhodujúcimi faktormi, uplatnenými pri výbere databáz bola dostupnosť obrázkov vo vysokom rozlíšení a požiadavka, aby obsahovali tvár zachytenú spredu. Taktiež bol dôraz na fakt, aby obrázky obsahovali jedincov rôzneho pohlavia, rôznych vekových kategórií či odlišných rás. Prvou z použitých databáz je voľne dostupná SiblingsDB, kým druhou je interná databáza VUT FIT nazvaná STRATA EBD - Irises. Celkové výsledky testovania boli veľmi pozitívne. Timm & Barth algoritmus dokáže spoľahlivo nájsť pozíciu zrenice, ale nie jej presné centrum. Úspešne však našiel zrenicu v 100% testovaných obrázkov. Po druhé, algoritmus Terence Brouns zvládne nájsť presný stred zrenice, ale zároveň existujú prípady, kedy to nie je schopný urobiť. Problém nastal v 10,3% testovaných obrázkov. Hlavnou príčinou bolo výrazné zníženie rozlíšenia obrázkov počas ich prispôsobenia, aby vyhovovali účelom platformy. Implementovaný systém dokáže spracovať jednu snímku v priemere za 1189 ms. Pri testovaní platformy boli použité 4 rôzne kamery, nakoľko sa ukázalo, že voľba kamery ma veľký vplyv na dosiahnuté výsledky. Presnosť sa tak pohybuje od 53.8% pri použití kamery s najhoršími parametrami, až po 97.2% pri použití kamery s vysokým rozlíšením a automatickým zaostrovaním. Tieto výsledky sú uspokojivé, nakoľko platforma bude disponovať vo výslednej verzií ešte kvalitnejšou kamerou. Ovládanie platformy taktiež počas testovania fungovalo spoľahlivo.

Z výsledkov vyplýva, že implementovaný systém môže byť úspešne využitý pre riadenie polohovacej platformy pre očné centrum. Ďalšie zlepšenie dátových sád či úprava niektorých parametrov by mohli viesť k reálnemu použitiu.

Control of Positionable Platform for Eye Centering in Image

Declaration

Hereby I declare that I have written this master's thesis on my own under the supervision of Prof. Ing., Dipl.-Ing. Martin Drahanský, Ph.D. and I have acknowledged all sources I have used while writing this thesis.

.....
Patrik Magdolen
July 30, 2018

Acknowledgements

I would like to thank Prof. Ing., Dipl.-Ing. Martin Drahanský, Ph.D., for supervising this thesis and providing lot of valuable advice, which contributed to this work. I would also like to thank my family, who supported me and helped me throughout my studies.

Contents

1 Introduction **3**

2 Human eye - anatomy and scanning **4**
2.1 Human eye . . . 4
2.2 Scanning devices . . . 5
2.2.1 Slit lamp . . . 6
2.2.2 Ophthalmoscope . . . 6
2.2.3 Fundus camera . . . 7
2.2.4 Scanning Laser Ophthalmoscope . . . 9
2.3 Developed platform . . . 10
2.3.1 Scanning process . . . 11

3 Eye detection and centralisation of pupil **12**
3.1 Eye centres localisation . . . 12
3.1.1 Feature-based methods . . . 13
3.1.2 Model-based methods . . . 13
3.1.3 Hybrid methods . . . 14
3.2 Algorithm Timm & Barth . . . 14
3.2.1 Prior knowledge and postprocessing . . . 16
3.2.2 Evaluation and comparison . . . 16
3.3 Terence Brouns's algorithm . . . 17
3.3.1 Methods . . . 17
3.3.2 Feature value prediction . . . 18
3.3.3 Search area . . . 21
3.3.4 Approximate detection . . . 22
3.3.5 Canny edge detection . . . 22
3.3.6 Morphological operation . . . 22
3.3.7 Edge selection . . . 23
3.3.8 Edge classifcation . . . 23
3.3.9 Edge segmentation . . . 24
3.3.10 Ellipse fitting . . . 27
3.4 Conclusion . . . 29

4 Design and implementation **30**
4.1 Implementation details . . . 30
4.2 System structure . . . 30
4.3 Program arguments . . . 31
4.4 Camera capture . . . 32

4.5 Specular highlights removal . . . 32
4.6 Implementation of Timm & Barth's algorithm . . . 33
4.6.1 Mask enhancement . . . 35
4.7 Implementation of Terence Brouns's algorithm . . . 36
4.8 Platform control . . . 38
4.8.1 Calculating the distance . . . 38
4.8.2 Details of movement commands . . . 38
4.8.3 Platform functions . . . 39

5 Testing & Evaluation **41**
5.1 Verification method . . . 41
5.1.1 Data sets . . . 41
5.2 Database testing . . . 41
5.3 Platform testing . . . 43
5.4 Time complexity . . . 46
5.5 Evaluation of functionality . . . 46
5.6 Further work . . . 47

6 Conclusion **48**

Bibliography **49**

A CD Content **53**

B Program usage **54**

Chapter 1

Introduction

Importance of biometrics as method of authentication and identification grew during the end of 20^{th} century and it became de facto mainstream in daily life when the first phones with fingerprint sensor were introduced to the market in 2013. Another powerful and reliable person identification methods include iris recognition and retina scanning. It can be used in many commercial sectors to protect malicious access or identify right persons with incredibly high accuracy. [13]

Of all the possible disabilities, loss of sight has the greatest impact on everyday life. This impact is greater than memory, voice, or hearing loss. This is because 80% of all sensory information that the human brain receives comes through vision. This is how the eye becomes the most important sensory organ. One person may be surprised that the loss of vision is not normal for the ageing process. But even more interesting is the fact that most cases of vision loss can be prevented by early detection of the disease and its treatment. Visiting doctors can be unpleasant to some people, not only because of the fear of examination but also because of the time-span. This can be at least partially eliminated by the development of automated devices, which could almost without the intervention of a doctor evaluate the condition of the patient. These devices can scan the human eye and its parts, or even diagnose the state of the patient's eye. [13]

For purposes of iris recognition and retina scanning, specialised hardware is being used with near infrared illumination to acquire precise position of iris. These solutions are often massive and expensive. Due to the rapid development of hardware in particular, it is possible to implement these systems on non-specialised devices, such as a single-board computer along with a web cam. In my thesis I covered the topic of eye centring in image. I focused on localisation of human eye pupil using direct computation. Such system could be then used by a device which is currently being developed by our faculty. This ophthalmologic device for automatic capturing of eye retina and iris has two possible ways of utilisation - either for biometric purposes or for medical purposes as supporting diagnostic device. It consists of lighting equipment, an adaptive optical system, an image capturing unit situated on position-able platform and requires alignment of the axis of the eye and the camera for its function.

Second chapter describes basic anatomy of eye, required for understanding methods described further. Chapter three contains detailed description of methods of eye detection and centralisation of pupil. Implementation of selected methods and description of program functionality is described in chapter three. Lastly, chapter four contains testing of these functions and their evaluation.

Chapter 2

Human eye - anatomy and scanning

In this chapter, at first the anatomy of the eye will be presented. The second part introduces commonly used scanning methods and scanning devices. Lastly, this chapter includes a detailed description of the platform being developed. These fundamentals are necessity in order to implement the algorithm for eye centre localisation.

2.1 Human eye

The eye has the function of the sense organ and attributes the most important task to all senses of man. It is a sensory analyser that allows us to receive information from the outside environment and provides us with 80 to 85 percent of the world's information. It provides information about shapes, colours, sizes and also the movement of various items placed in areas. It also has an important role to play in maintaining biorhythm. Creating an image and vision itself are complex processes. The origin of the image is the transformation of the light energy absorbed in the retinal photo-receptors of the electrical signal that is transmitted by the visual up to the vision centre located in the cortex of the main lobe of the brain. Eye is a sensory organ consisting of 3 layers: [12]

- *The outer layer* - the layer of connective tissue, which forms the sclera and the cornea,
- *the middle layer* - vascular layer, consisting of iris, ciliary body and choroid,
- *the inner layer* - it is a nerve layer that is made of a retina.

This specific organ is divided into three areas: the anterior chamber of the eye, the posterior eye chamber and vitreous. Each part of the eye has a specific and irreplaceable function. The sclera is white and opaque, and its surface covers the link. The cornea, which is transparent, allows transitions of light rays into the inside of the eye, and helps them concentrate on the retina. The iris has a function of the membrane, that regulates how much light gets to the eye's pupil. The diameter of the pot is under the function of the iris muscles, that are controlled by the autonomic nervous system. The ciliary, or wrinkle body, is also the producer of the components of the ventricular fluid in the eye and, at the same time, it also contains muscle, that controls the shape and curvature of the lens. Choroid is a dense vascular knot surrounding the retina. Its role is to deliver the necessary

nutrients to the structure of the eye. The role of the retina - the cornea, which is formed by nervous tissue, is, by means of complex biochemical processes, to change the sun rays on it, that impinge on the nerve signal, which is then transmitted further. The retinal signal is transmitted through the optic (visual) nerve out of the eye and gets into different parts of the brain where it is subsequently subject to processing. The optic nerve is made up of more than a million nerve fibbers. First section processing takes place in the eyeball, the second in the skull, where the visual nerves of both sides cross and combine. The lens provides precise focusing of the rays on the retina. An important feature of the lens is that it is able to accommodate itself. This means that it is able to change its own shape and allows us to see objects near us. [30] [12]

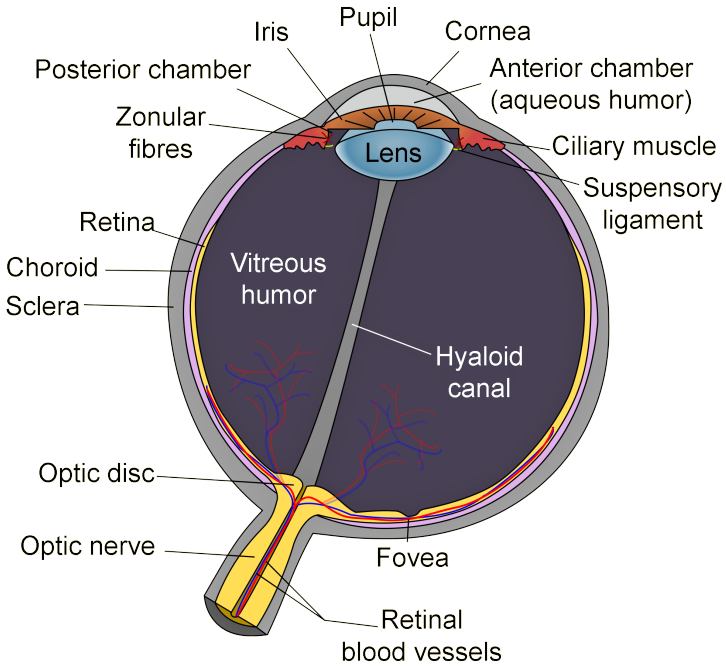

Figure 2.1: Schematic diagram of the human eye [10].

Ventricular fluid produced by the ciliary body, fills the front and rear eye chambers. These two chambers are communicating with each other through the pupil. The function of ventricular fluid is to provide nutrition for the surrounding parts of the eye, especially cornea and lens. The vitreous body, which occupies the largest space in the eye, is bounded from the front by a lens, and from behind it presses the retina. The vitreous body is formed by the glassy fluid with gel consistency. [30] [28]

2.2 Scanning devices

The basic ophthalmoscopic examination methods of the anterior segment and posterior segment of the eye include direct and indirect ophthalmoscopy, whilst nowadays, examination using a slit lamp is the most common. In the past few decades, the use of light has played an important role in revealing structural and functional information from the human retina

in a non-destructive and non-invasive manner. Ophthalmic optics, as an active research area, has been expanding steadily, providing scientists and doctors with priceless multidisciplinary information and enabling new diagnostic and therapeutic methods. Nowadays, digital imaging technology is predominant in this specialised field, using the most common CCD chips to create images with a resolution of more than 18 megapixels, or high resolution video at a scanning speed of 15 to 30 frames per second. Beam splitters are embedded between the lens and the magnification changer and its distribution ratio determines the ratio between the luminous flux diverted to the video adapter and the flow going to the patient's eye. [3]

2.2.1 Slit lamp

The slit lamp is an instrument consisting of a high-intensity light source, which can be focused to shine a thin sheet of light into the eye. The slit lamp facilitates an examination of the anterior segment and posterior segment of the human eye. This includes the eyelid, sclera, conjunctiva, iris, natural crystalline lens, and cornea. Slit lamps are a combination of a binocular microscope and a lighting unit. Based on the location of the light source, slit lamps are divided into two basic types. The first type is Haag Streit lamp, with the upper light source, where the rays of light are reflected in the eye examined by the mirror. The other type is Zeiss lamp, with a lower source, which uses a prism to bend the light to the eye. Biomicroscopy of the fundus with the slit lamp is ultimately the result of melding the two most important examination techniques in clinical ophthalmology, namely biomicroscopy with the slit lamp and fundoscopy. [11]

Figure 2.2: Side view of a slit lamp machine. [17]

2.2.2 Ophthalmoscope

Ophthalmoscope, instrument for inspecting the interior of the eye. The ophthalmoscope generally is considered to have been invented in 1851 by the German physiologist Hermann von Helmholtz, though it is sometimes credited to English mathematician and inventor

Charles Babbage, who in 1847 developed an instrument thought to resemble the ophthalmoscope. The device consists of a strong light that can be directed into the eye by a small mirror or prism. The light reflects off the retina and back through a small hole in the ophthalmoscope, through which the examiner sees a nonstereoscopic magnified image of the structures at the back of the eye, including the optic disk, retina, retinal blood vessels, macula, and choroid. The ophthalmoscope is particularly useful as a screening tool for various ocular diseases, such as diabetic retinopathy. The pupil is a hole through which the eye's interior will be viewed. Opening the pupil wider is a simple and effective way of seeing the structures behind it better. Therefore, dilation of the pupil is often accomplished with medicated eye drops before funduscopy. However, although dilated fundus examination is ideal, undilated examination is more convenient, and it is the most common type in primary care. [18]

Figure 2.3: Ophthalmoscope and otoscope combination. [9]

2.2.3 Fundus camera

The fundus camera, sometimes also called the retinal camera, is a special and highly sophisticated device for displaying the posterior segment of the eye i.e. fundus. Fundus photography documents the retina, the neurosensory tissue in our eyes, which translates the optical images we see into the electrical impulses sent to the brain. Specialised fundus cameras that consist of an intricate microscope attached to a flash enabled camera are used in fundus photography. The patient sits at the fundus camera with their chin in a chin rest and their forehead against the bar. An ophthalmic photographer focuses and aligns the fundus camera. A flash fires as the photographer presses the shutter release, creating a fundus photograph like the picture in figure 2.4. It works on the principle of indirect ophthalmoscopy, when a source of primary white light is embedded inside the instrument. Light can be modified by different types of filters, and the optical system is focused on the patient's eye, where it bounces off the retina and returns to the fundus camera lens. Here, again, using another optical system, the image is modified and then digitised using CCD chips. Fundus cameras are described by the angle of view - the optical angle of acceptance

of the lens. The normal scan range is between 45° and 50°, and the software can create a panoramic view of the eye background. Since this angle is insufficient for scanning edges of retina, it is usually necessary for a patient to look in different directions and multiple different photographs are then combined into final fundus photograph. There are also special fundus cameras, called the RatCam Shuttle, which can show the eye at 130° at a time. Thanks to sophisticated technology, it is not necessary to investigate the eye background in mydriasis, which is why, many of today's cameras offer a nonmydritic mode, when the quality images can be achieved with skoptical conditions and a pupil width of 3.5 mm. In analogy with the indirect ophthalmoscope, the objective lens forms a real intermediate image of the illuminated fundus in front of a pinhole mirror. Behind the pinhole mirror, a second intermediate image is formed by the main objective lens. [26] [5]

Figure 2.4: Non-mydriatic auto fundus camera AFC-330. [2]

Fundus photographs are visual records which document the current ophthalmoscopic appearance of a patient's central and peripheral retina. Main interest is the optic nerve photography, allowing the evaluation of structural relationships within the nerve. It also allows the practitioner to examine fine details not easily seen on examination, as well as evolution of such changes over time. They allow the physician to further study a patient's retina, to identify retinal changes on follow-up, or to review a patient's retinal findings with a colleague. Fundus photographs are routinely ordered in a wide variety of ophthalmic conditions. For example, glaucoma (increased pressure in the eye) can damage the optic nerve over time. Using serial photographs, the physician studies subtle changes in the optic nerve and then recommends the appropriate therapy. Fundus photography is also used to document the characteristics of diabetic retinopathy (damage to the retina from diabetes), such as macular edema and microaneurysms. This is because retinal details may be easier to visualize in stereoscopic fundus photographs as opposed to direct examination. Fundus photography is also used to help interpret fluorescein angiography because certain retinal landmarks visible in fundus photography are not visible on a fluorescein angiogram. [26] [32] [31]

Figure 2.5: Normal fundus photographs of the right eye. [15]

The cost of fundus photography continues to be significantly lower than the newer techniques based on retinal scanning. Its main advantages are the easy interpretation, full colour, better detection of disc hemorages, peripapillary atrophy etc. Disadvantages include lack of quantitative description and hence inter-observer variability, need of highest photographic quality, and difficult serial comparison because of limited ability to detect subtle changes with a photograph. [26]

2.2.4 Scanning Laser Ophthalmoscope

In the scanning laser ophthalmoscope, a narrow (ca. 1 mm) laser beam of safe intensity traverses the optical axis to a single point (ca. 10 μm diameter) on the retina, resulting in comfort for the patient. The fundus image was produced by scanning the laser over the retina in a raster pattern, detecting the signal from each point scanned, to produce a digital image. Beam deflection was achieved by a combination of two galvanometer scanners – one slow vertical scanner ($\sim$ 60 Hz), and one fast horizontal scanner ($\sim$ 15 kHz). As a method used to image the retina with a high degree of spatial sensitivity, it is helpful in the diagnosis of glaucoma, macular degeneration, and other retinal disorders. It has further been combined with adaptive optics technology to provide sharper images of the retina. Compared to fundus imaging, scanning technologies are generally applied to smaller portions of the retina, but allow higher resolution and depth-penetration. While it is able to image the retina in real time, it has issues with reflections from eye astigmatism and the cornea. Furthermore, eye movements can confound the data from scanning laser ophthalmoscope. Additionally, adaptive optics scanning laser ophthalmoscopy is a technique used to measure living retinal cells. It utilises adaptive optics to remove optical aberrations from images obtained from scanning laser ophthalmoscopy of the retina. [40]

Figure 2.6: Scanning laser ophthalmoscope NIDEK F-10. [1]

2.3 Developed platform

The device for acquirement and recognition of human eye characteristics was created by researchers from Faculty of Information Technology. Preview of this device can be seen in figure 2.7. The device is able to scan two characteristics – eye iris and eye retina. The main principle is based on optics, camera systems and special rotation carousel inside the device, which enable to acquire both eye characteristics in one compact device. The optical system is responsible for image capture and allows proper focusing of the iris and retina of the eye. The integrated light source works both with visible and infrared light. The entire optical system is connected to a position-able 3D platform that is controlled on the basis of the image from the camera system. [14]

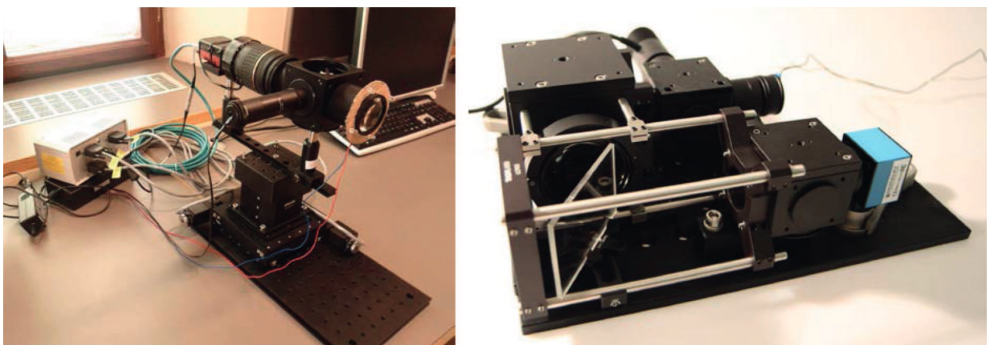

Figure 2.7: Display of laboratory device.

The device may be used for recognition of a physical identity of the human in security applications. This small and compact solution can be used for verification of identity, offering highly reliable results. Another advantages include a non-invasive procedure, very high resistance against spoof attacks and ease of use, making it suitable for large scale databases or user groups. Additionally, the device may be used as an ophthalmologic medical device. It will enable a medical doctor to scan eye iris and retina in one device. The device comes with software based on an expert system that will assist the medical

doctor to make a diagnosis and to recognise an illness in the eye. One of the features is the possibility to teach the expert system new diseases, so that the device recognises the given disease automatically next time. The patient data can be stored and checked against new images, i.e. it will be possible to determine the disease progress.

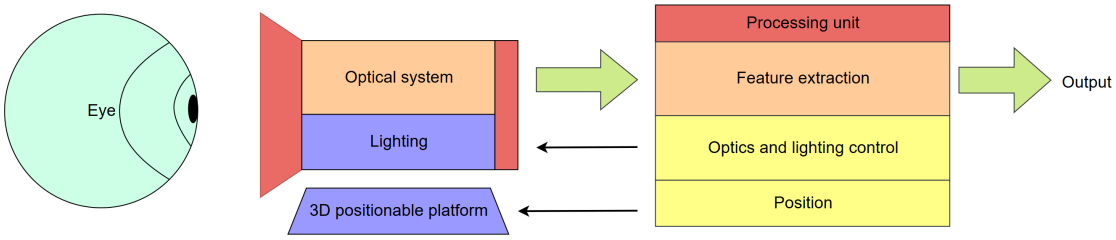

Figure 2.8: Block scheme of scanning device.

Block scheme of this device can be seen in figure 2.8. The central component of the system is the signal processing unit. This unit acquires and processes images from an optical system. After the image is quickly reviewed, this unit sends an image to an output device such as a computer. The basic modules of the unit are the flag extractor and template generator. Another part of the unit is a model of optical control and light intensity based on feedback from the camera system. The positioning model ensures alignment of the optical axis of the camera system with the optical axis of the scanned eye using the positionable platform. The information about position is again retrieved from the camera system feedback.

The positioning algorithm adjusts the optical axis of the camera system according to the optical axis of the eye. Through the adjustable platform in three axes, it is possible to increase user friendliness, as well as greatly reduce the time for the scanning itself. Optical system parameters are designed to meet the requirements resulting from physiological properties of the human eye, but also from medical examination of the eye, or suitability for biometric recognition of persons.

2.3.1 Scanning process

Process of acquirement of eye iris and eye retina consists of the following steps:

1. The user has to be in contact with the device for the acquirement to commence. The distance of the user is automatically measured, i.e. the device knows the correct position of users head. After the correct positioning of the head, the correct position of the eye is searched and the user is navigated to reach the appropriate position.

2. Then, the scanning process of eye iris can start. The eye is illuminated with infrared light and the iris is scanned.

3. Continuously, the device is set to the mode of eye retina scanning. The optical axis of the eye has to be set to a perpendicular position, then the eye background is illuminated with visible light and the retina image is acquired.

Chapter 3

Eye detection and centralisation of pupil

Robust eye detection and tracking algorithms are crucial for a wide variety of disciplines, including human computer interaction, medical research, optometry, biometry, marketing research, and automotive design. In this chapter, at first, analysis of various approaches to eye centres localisation technology with focus on examples and their comparison is presented. Later, based on this analysis, the final method itself will be proposed.

3.1 Eye centres localisation

Over the past decades eye localisation and tracking has been one of the most active research areas in the field of computer vision, mainly due to its application to a wide variety of research fields. The organisation of all the eye localisation methods into a single and general taxonomy proves to be a challenging task. However, these methods can be broadly classified into two categories [20]:

- *AELS - active eye-localization systems*,
- *PELS - passive eye-localization systems.*

In recent years, the *AELS* have improved significantly, and are beginning to play an important role in the field of human–computer interaction. However, a typical *AELS* is generally expensive since it often requires auxiliary equipment, such as a head-mounted device or an infrared camera [39]. These systems use the image coordinates of the pupil and corneal reflection [20]. Unlike the *AELS*, the *PELS* attempts to directly obtain information about the iris center location based only on the images supplied from a camera video stream only [20]. Fortunately, further advancements have been made recently. Due to the increased speed of computer processors, improved computer vision techniques and the availability of high performance digital cameras, the most desired type localisation output, which estimates the coordinates of the user's gaze, can be easily implemented. Therefore, in this study, we investigate localisation methods, specifically focusing on those that can operate when only common digital cameras are available. Furthermore, we discuss approaches to improve the robustness and accuracy of this technology in detail. [29] [23]

Despite active research and significant progress in the last three decades, eye detection and tracking remains a very challenging task owing to the individuality of human eyes, occlusion of the eye by the eyelids, variability in scale, location, reflectivity, head pose,

cases of open/closed eyes, light conditions, etc. Various eye localisation methods have been proposed in the literature, and can be roughly divided into these three classes: [29]

- *Feature-based methods*,
- *model-based methods*,
- *hybrid methods*.

3.1.1 Feature-based methods

First class of localisation methods are *feature-based methods*, which use pertinent eye properties to detect candidate eye centres. The pertinent features may be edges, eye corners, or points, which are selected based on specific filter responses.

For example, the method proposed by Wang [37] exploit the fact that the outer boundary of the iris is a circle. With a fully calibrated camera, its elliptical image can be back-projected into the 3D space yielding two possible circles. To disambiguate, the solution is found by making use of anthropomorphic knowledge of the structure of the eyeball. In comparison, Asadifard and Shanbezadeh [4] proposed an adaptive method based on the cumulative density function, which filters the eye image to determine the pixel values that are likely to belong to the pupil. In work of Zheng and Yang [42], the eye's features, including the pupil centre and radius, eye corners, and eyelid contours, are detected from frontal colour facial images by integrating colour information, Gabor features, and the mutual localisation relationship between different features. Valenti and Gevers [34] propose an alternative voting scheme that uses isophote properties, like curves connecting points of equal intensity, in the intensity image to detect the location of the eyes. However, since this method relies on maxima in the feature space, it may detect the eyebrow or eye corners instead of the iris centre when the number of features in the eye region is insufficient.

The advantages of the *feature-based methods* are that they are robust to shape and scale changes and have lower computational complexity, without requiring any learning or model fitting. When the extracted features are not confused by noise or surrounding features, the resultant eye location can be very accurate. Nevertheless, the limitations of these methods are revealed in certain difficult scenarios, for example, when the user rotates his eyeball to either corner, when the user almost closes his eyes, or when there is occlusion due to eyelids or eye corners. The detected features might often be incorrect and, therefore, these methods frequently fail to accurately estimate the eye centres. [21]

3.1.2 Model-based methods

Second category consists of *model-based methods*, which employ a prior model of eye holistic appearance and surrounding structures, even the face, and often use classification of a set of features or the fitting of a learned model to estimate the location of the eyes.

Examples include Moriyama's [24] use of the generative eye region model, which can distinguish eye components by parameterising the fine structure and motion of the eye. The system first registers the eye model to the input in a particular frame and individualises it by adjusting the structure parameters. The system then tracks motion of the eye by estimating the motion parameters across the entire image sequence. On the other hand, Hamouz [19] proposed a method that search for ten feature points on the face using Gabor filters, apply a triplet of such corresponding features to define an affine transformation, provide a face hypothesis and, finally, verify the remaining configurations using two cascaded Support

Vector Machine classifiers. The method by Reale [27] creates a 3-D iris disk by mapping both the iris centre and iris contour points to the eyeball sphere, then fits a circle to the iris to find the optimal eye ball rotation. Wang proposed a method [38] that first applies statistically learned non-parametric discriminant features to characterise eye patterns, then determines probabilistic classifiers to separate eye and non-eye features and, finally, multiple classifiers are combined in AdaBoost to form a robust and accurate eye detector.

By using both the global facial appearance and computer learning, *model-based methods* have the advantage of being very robust and well-suited for precise detection of overall eye location. However, the model-based methods have a drawback in that they usually require a large amount of training data to be collected and the model parameters need to be iteratively or manually adjusted, therefore, *model-based methods* are inappropriate for practical applications. Moreover, these methods usually fail to locate the eye centers when they are faced with subtle eye-center movements. [21]

3.1.3 Hybrid methods

Lastly, *hybrid methods* aimed at combining the advantages of both *feature-based* and *model-based methods* within a single system have been developed to overcome the respective shortcomings of each approach.

For instance, Huang and Wechsler [22] suggest an adaptive hybrid eye localisation approach, using a consensus between navigation routines encoded as finite-state automata that explore the facial landscape to derive a saliency map, which is developed using genetic algorithms. These salient regions are then classified as eyes using genetically evolved decision trees. Wang [36] treat faces as an image topographic manifold and use a terrain-classification procedure on the topographic manifold to generate a terrain map. In order to select proper eye pairs from the topographic manifold candidates, a SVM based on the Bhattacharyya kernel is applied. Valenti and Gevers [35] propose a hybrid scheme that uses isophote properties in the intensity image to detect eye location, and utilize mean shift and machine learning to overcome problems that arise in certain lighting conditions of due to occlusions from the eyelids.

3.2 Algorithm Timm & Barth

This whole chapter simplifies text from article [33]. Authors Fabian Timm and Erhardt Barth from institute for Neuro- and Bioinformatics, University of Lubeck, Germany proposed feature-based approach for eye centre localisation that can efficiently and accurately locate and track eye centres in low resolution images and videos by using image gradients. A simple objective function is derived, which only consists of dot products. For every pixel, the squared dot product is computed between the displacement vector of a centre candidate and the image gradient. The position of the maximum then corresponds to the position where most image gradients intersect and thus to the eye's centre. This method yields low computational complexity and is invariant to rotation and linear changes in illumination.

Their approach makes the following contributions:

1. A novel approach for eye centre localisation, which defines the centre of a circular pattern as the location where most of the image gradients intersect. Therefore, they derive a mathematical function that reaches its maximum at the centre of the circular pattern. By using this mathematical formulation a fast iterative scheme can be derived.

2. Incorporation of prior knowledge about the eye appearance and increase the robustness.

3. Appliance of simple postprocessing techniques to reduce problems that arise in the presence of glasses, reflections inside glasses, or prominent eyebrows. Furthermore, evaluation of the accuracy and the robustness to changes in lighting, contrast, and background by using the very challenging BioID database. The obtained results are extensively compared with state of the art methods for eye centre localisation.

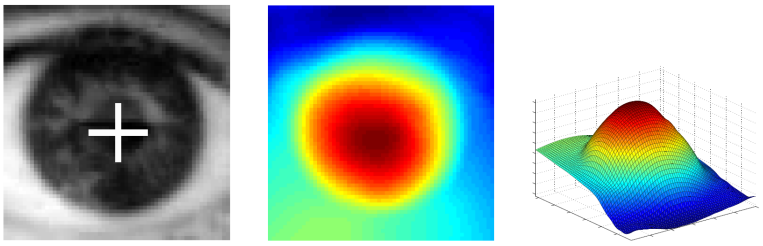

Figure 3.1: Evaluation for an exemplary pupil [33].

Geometrically, the centre of a circular object can be detected by analysing the vector field of image gradients. In their work, Timm and Barth analyse the vector field of image gradients and derive a novel mathematical formulation of the vector field characteristics. Therefore, mathematical description of the relationship between a possible centre and the orientations of all image gradients is created. Let c be a possible centre and g_i the gradient vector at position x_i. Then, the normalised displacement vector d_i should have the same orientation, except for the sign, as the gradient g_i. If we use the vector field of image gradients, we can exploit this vector field by computing the dot products between the normalised displacement vectors related to a fixed centre and the gradient vectors g_i. The optimal centre c^* of a circular object in an image with pixel positions $x_i, i \in \{1, ..., N\}$, is then given by

$$c^* = argmax \left\{ \frac{1}{N} \sum_{i=1}^{N} \left(d_i^T g_i \right)^2 \right\} \tag{3.1}$$

$$d_i = \frac{x_i - c}{\|x_i - c\|_2} \tag{3.2}$$

$$\forall i : \|g_i\|_2 = 1 \tag{3.3}$$

The displacement vectors d_i are scaled to unit length in order to obtain an equal weight for all pixel positions. In order to improve robustness to linear changes in lighting and contrast the gradient vectors should also be scaled to unit length. An example evaluation of the sum of dot products for different centres is shown in Figure 3.1, where the objective function yields a strong maximum at the centre of the pupil.

Computational complexity can be decreased by considering only gradient vectors with a significant magnitude, i.e. ignoring gradients in homogeneous regions. In order to obtain the image gradients, we compute the partial derivatives

$$g_i = \left(\partial I \left(x_i, y_i \right) / \partial x_i, \partial I \left(x_i, y_i \right) / \partial y_i \right)^T \tag{3.4}$$

but other methods for computing image gradients will not change the behaviour of the objective function significantly.

3.2.1 Prior knowledge and postprocessing

Under some conditions, the maximum is not well defined, or there are local maximums that lead to wrong centre estimates. For example, dominant eyelids and eyelashes or wrinkles in combination with a low contrast between iris and sclera can lead to wrong estimates. Therefore, they propose to incorporate prior knowledge about the eye in order to increase robustness. Since the pupil is usually dark compared to sclera and skin, a weight w_c is applied for each possible centre c such that dark centres are more likely than bright centres. Integrating this into the objective function leads to:

$$argmax \frac{1}{N} \sum_{i=1}^{N} w_c \left(d_i^T g_i\right)^2 \tag{3.5}$$

where $w_c = I^*(c_x, c_y)$ is the grey value at (c_x, c_y) of the smoothed and inverted input image I^*. The image needs to be smoothed by a Gaussian filter, in order to avoid problems that arise due to bright outliers such as reflections of glasses. The values of the new objective function is rather insensitive to changes in the parameters of the low-pass filter.

The proposed summation of weighted squared dot products yields accurate results if the image contains the eye. However, the rough eye regions sometimes also contain other structures such as hair, eyebrows, or glasses. Especially, hair and strong reflections in glasses show significant image gradients that do not have the same orientation as the image gradients of the pupil and the iris, hence the estimation of the eye centres might be wrong. Therefore, postprocessing step is used in order to overcome these problems. A threshold is applied on the objective function, based on the maximum value, and remove all remaining values that are connected to one of the image borders. Then, the maximum of the remaining values is determined and its position is used as centre estimate. Based on authors experiments, the value of this threshold does not have a significant influence on the centre estimates and thus Timm and Barth suggest to set this threshold to 90% of the overall maximum.

3.2.2 Evaluation and comparison

Authors used BioID database for evaluation, as the most challenging set of images for eye centre localisation. The database consists of 1521 grey level images of 23 different subjects and has been taken in different locations and at different day times, which result in variable illumination conditions comparable to outdoor scenes. In addition to the changes in illumination, the position of the subjects change as well as their pose. Moreover, several subjects wear glasses and some subjects have curled hair near to the eye centres. In few images the eyes are even completely hidden by strong reflections on the glasses. Compared to several state of the art methods, this method yields a very high accuracy over several scenarios when pupil localisation, iris localisation, and overall eye localisation, is evaluated. It can be observed that this approach yields accurate centre estimations not only for images containing dominant pupils, but also in the presence of glasses, shadows, low contrast, or strands of hair. This demonstrates the robustness and proves that this algorithm can successfully deal with several severe problems that arise in realistic scenarios. Furthermore,

it can be applied to several real-time applications that require a high accuracy such as eye tracking or medical imaging analysis, while having relatively low performance requirements.

3.3 Terence Brouns's algorithm

This whole chapter simplifies text from article [6]. Author Terence Brouns from Radbound University, the Netherlands presents a pupil detection method that obtains reliable predictions through recursive estimation about certain pupil characteristics in successive camera frames. These predictions are subsequently used to carry out novel image segmentation and classification routines to improve pupil detection performance. Based on results from hand-labelled eye images, their approach was found to have a greater detection rate, accuracy and speed compared to other recently published pupil detection algorithms. Only drawback of this methods is that it only works on images of eye itself.

3.3.1 Methods

The pupil detection algorithm, which is illustrated in figure 3.2, works by performing a number of distinct processing tasks in the following order:

1. Receive predicted pupil characteristics for current frame
2. Crop image to smaller search area depending on predicted size, shape and position of pupil
3. Update predicted pupil position following object recognition of the pupil through Haar-like feature detection, after removing corneal reflection interference
4. Detect all object boundaries, i.e. edges, in area around new position estimate by identifying brightness discontinuities with Canny edge detection
5. Select sub-set of detected edges that are at an acceptable distance from the position estimate
6. Thin Canny edges to minimum thickness using morphological operations
7. Segment edges at transition point between distinct features in the image
8. Classify edges into two classes: pupil contour edges and non-pupil contour edges
9. Fit one or more ellipses on edges categorized as pupil contour edges
10. Choose most optimal fit based on pupil characteristic predictions
11. Calculate new predictions for next frame

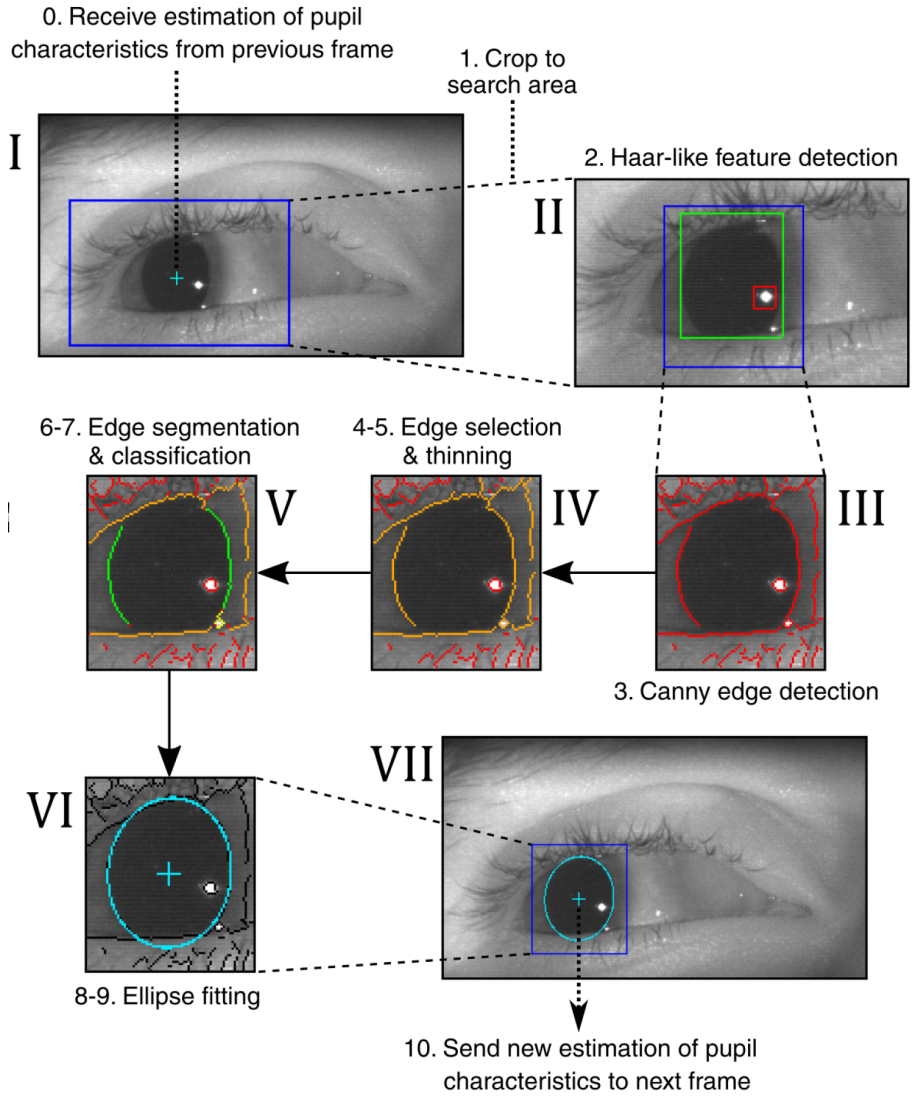

Figure 3.2: Example of an image being sent through the pupil detection pipeline.

3.3.2 Feature value prediction

The main strength of the pupil detection algorithm is drawn from the use of predictions that are made about certain characteristics of the pupil in each camera frame. A popular type of recursive estimator is the Kalman filter, which has been extensively used in computer vision research. However, because we are interested in estimating many different variables, the use of the Kalman filter is made difficult due to the exponentially increased complexity of setting the measurement and process noise covariances, which are integral to the functionality of the filter. Therefore, algorithm uses a more basic recursive estimation method, which may lack the accuracy of the Kalman filter, but requires fewer parameters to be set. This recursive estimator is used to keep track of the pupil features that are listed in table 3.1.

Estimated features	Symbol	Description
Position	$\hat{f}$	Pupil centre position in Cartesian coordinate
Circumference	$\hat{C}$	Circumference of pupil-iris boundary
Aspect ratio	$\hat{AR}$	Ratio between pupil major and minor axes
Width	$\hat{W}$	Width of pupil bounding box
Height	$\hat{H}$	Height of pupil bounding box
Angle	$\hat{\theta}$	Pupil rotation angle
Brightness/intensity	$\hat{I}$	Grey-scale value of the inner pupil-iris boundary
Radial gradient	$\hat{G}_r$	Radial image gradient of the pupil-iris boundary
Curvature	$\hat{\kappa}$	Signed curvature of pupil-iris boundary

Table 3.1: Pupil features.

After the pupil has been detected in a particular camera frame, each one of these features is updated with the following general formula:

$$\hat{f}_{n+1} = \hat{f}_n + \alpha \Delta f_n + c_n p_n \tag{3.6}$$

Here, f_n is the predicted feature value for the current frame n. This value is updated to obtain the prediction for the next frame, f_{n+1}, by adding the difference Δf_n between the measured and the predicted value (the prediction error), plus a momentum term, p_n. In every frame where we detect the pupil, we will possess a measured feature value f_n, which is used to calculate Δf_n through:

$$f_{n+1} = f_n - \hat{f}_n \tag{3.7}$$

The value of Δf_n is modified by the gain factor α in equation 3.6, which is a constant value between 0 and 1. The larger the gain, the more the estimation is based on the current measurement f_n. The value of α depends on the feature in question. A feature such as pupil position updates relatively quickly, as it can shift rapidly on short timescales. On the other hand, significant changes in the pupil's size and shape generally occur on longer timescales, allowing for a smaller gain, which helps reduce the influence of noise. Using only Δf_n to update f_n would fail when the feature value monotonically increases, because prediction will lag behind. For this reason, a momentum term p n has been added in equation 1, where p_n is updated in every frame by:

$$p_{n+1} = p_n + \alpha(\Delta f_n - p_n) \tag{3.8}$$

What p_n tries to approximate is the prediction error Δf_n of the feature. Ideally, p_n causes Δf_n to drop to zero, resulting in accurate predictions. Furthermore, p_n is modified by an additional factor c_n, which gives a measure of the certainty of a prediction and it's value is between 0 (low) and 1 (high). This factor is added to the equation to avoid erratic behaviour when little information about the pupil is available. The value for c_n is altered according to the rate of change of the feature. If the rate falls within an acceptable range, c_n increases, otherwise it decreases. This is given by:

$$\Delta c_n = 1 - \frac{2}{1 + e^{k(\delta_n - \delta'_\theta)}} \tag{3.9}$$

The variable δ_n tell us how much the feature value has shifted from one frame to the next. The constant δ'_θ is an upper limit of typical δ_n values and is determined empirically.

If δ_n is below this threshold value of δ'_θ, the function will return a positive number, causing c_n to increase. Alternatively, c_n decreases when $\delta_n < \delta'_\theta$. The steepness k of the curve is determined by:

$$k = log \frac{\frac{1}{a} - 1}{b\delta'_\theta} \tag{3.10}$$

Where a is set to 0.99 and b is set to 0.50, which translates to the logistic function reaching 99% of its maximum value at $\delta_n = \frac{1}{2}\delta'_\theta$. These parameters are set in this way, because the certainty measure should not be biased towards ever smaller δ_n values, but should view all δ_n values that are physiologically feasible as equivalent. We update the certainty c n by first computing an intermediate value:

$$c'_{n+1} = c'_n + \alpha\alpha_c\Delta c_n \text{ with } c'_n = \begin{cases} 1 & \text{if } c'_n > 1 \\ 0 & \text{if } c'_n < 0 \end{cases} \tag{3.11}$$

Where δc_n is modified with the product of α and a new gain factor α_c. The certainty value used in equation 3.6 is calculated through another logistic function:

$$c_n = \frac{1}{1 + e^{-\tau(c'_n - \frac{1}{2})}} \tag{3.12}$$

The use of the logistic function has two purposes. First, the function bounds c_n in the 0 to 1 range. Second, the function imposes some latency for changes in certainty from the limits, which is controlled by the constant τ factor. As a consequence, multiple precise measurements are required before the certainty starts to significantly increase from its minimum value, or multiple imprecise or non-detections before certainty drops down from the maximum. Essentially, we create three different states the algorithm can attain: a focused state when certainty is high ($c \approx 1$), an exploratory state when it is low ($c \approx 0$), and a narrow transitional state in between. Both δ'_θ and δ''_θ need to be altered according to the frame-rate of the camera. The thresholds can be decreased when using a faster camera and should be increased with slower cameras. The various δ_n values are given by:

$$\delta_s = \sqrt{(x_n - \hat{x}_n^2) + (y_n - \hat{y}_n^2)} \tag{3.13}$$

$$\delta_C = \frac{|C_n - \hat{C}_n|}{max(C_n, \hat{C}_n)} \tag{3.14}$$

$$\delta_{AR} = |AR_n - \hat{AR}_n| \tag{3.15}$$

Up until this point, we have not dealt with the issue of non-detections, i.e. frames in which the algorithm has not been able to find the pupil. In the case of a non-detection, measures of f_n and δ_n are not available, which stops us from updating our feature predictions. We deal with this problem via certainty calculation by giving Δc_n the minimum value of –1 when a non-detection occurs. For the feature predictions, this issue is resolved by introducing the average feature variable $\bar{f}_n$, which represents a quantity that our prediction $\hat{f}_n$ can fall back on when no immediate information about the pupil is available. For position, we instead rely on the detection algorithm to supply an estimation of the position prediction $\hat{s}_n$ during non-detections. We update f_n during a non-detection according to:

$$\hat{f}_{n+1} = \hat{f}_n + \alpha_A\Delta\hat{f}_n + c_{A,n}p_n \tag{3.16}$$

$$\Delta\hat{f}_n = \bar{f}_n - \hat{f}_n \tag{3.17}$$

The momentum term p_n decays to zero by setting $\Delta\hat{f}_n$ to zero, leaving:

$$p_{n+1} = (1 - \alpha_A)p_n \tag{3.18}$$

Similarly to $\hat{f}_n$, we update $\bar{f}_n$ in every frame, but without using the momentum term:

$$\bar{f}_{n+1} = \bar{f}_n + \alpha_{mean}\Delta\bar{f}_n \tag{3.19}$$

$$\Delta\bar{f}_n = \hat{f}_n - \bar{f}_n \tag{3.20}$$

The gain factor of α mean should be much smaller than α_A in order for $\bar{f}_n$ to keep track of an overall mean of f_n. During a non-detection, $\bar{f}_n$ in equation 3.20 is replaced by a typical value that is initially assigned to $\hat{f}_n$ and $\bar{f}_n$ at the start of detection. The end result is that in every frame we have a prediction $\hat{f}$ for each type of feature, together with a measure c of how certain we are about that prediction. These quantities play a role in almost every part of the pupil detection algorithm.

3.3.3 Search area

The first application of feature predictions and certainties is in reducing the processing area for pupil detection. By recognising that the pupil can only translate and transform a limited amount between consecutive frames, we can narrow our search of the pupil to an area that is much smaller than the total size of the image. The width W_{AOI} and height H_{AOI} of this area of interest (AOI) still need to be at least as large as the predicted width $\hat{W}$ and height $\hat{H}$ of the pupil, thus we set:

$$W_{AOI} = \Delta L + \hat{W} \tag{3.21}$$

$$H_{AOI} = \Delta L + \hat{H} \tag{3.22}$$

Where ΔL is some additional length that is added to our prediction for the pupil size to ensure that the pupil falls wholly in the AOI. This length is determined by how much the pupil could potentially move and resize between two successive frames:

$$\Delta L = \frac{\hat{C}\delta_{\theta,C}}{\pi} + 2\delta_{\theta,s} \tag{3.23}$$

The variables $\delta_{\theta,s}$ and $\delta_{\theta,C}$ are derived from the threshold values δ''_θ in the following way:

$$\delta_{\theta,s} = (1 - c_S)(\delta_{max,s} - \delta''_{\theta,s}) + \delta''_{\theta,s} \tag{3.24}$$

$$\delta_{\theta,C} = (1 - c_A)(\delta_{max,C} - \delta''_{\theta,C}) + \delta''_{\theta,C} \tag{3.25}$$

The degree of certainty modifies the size of the AOI, with greater c values resulting in smaller processing areas, up to a minimum value of δ_θ and a maximum value of δ_{max}. These maximum values are a type of theoretical upper limit and are given by following equations, where W_{img} and H_{img} respectively are the width and height of the input image.

$$\delta_{max,s} = max(W_{img} - \hat{W}, H_{img} - \hat{H}) \tag{3.26}$$

$$\delta_{max,C} = max(\frac{C_{max} - \hat{C}}{C_{max}}, \frac{\hat{C} - C_{min}}{\hat{C}}) \tag{3.27}$$

3.3.4 Approximate detection

It is essential that a rough estimate of the pupil position is available for several parts of the algorithm. When c_S is low, $\hat{s}$ is re-evaluated by convolving the image with a Haar-like feature detector. This type of detector works by moving a Haar-like feature over the image and calculating the difference in the sum of all pixel values between the dark and light rectangular regions. This detection method is made computationally efficient by first calculating the integral image, which ensures $O(n)$ performance with respect to the number of input pixels. In this algorithm, vertical line feature is used to obtain approximate position of pupil, because it allows to better discriminate between pupil and eye lashes. This does mean, however, that a relatively bright feature can potentially mislead the feature detector if it happens to be flanked by an even brighter area. For this reason, both the intensity of the dark region as well as the contrast between dark and light regions are considered when calculating the Haar-like feature response, F_H:

$$F_H = -w_1\bar{i}_c + w_2(\frac{\bar{i}_l + \bar{i}_r}{2} - \bar{i}_c) \tag{3.28}$$

Where $\bar{i}$ is average pixel intensity. Both terms are modified by a weight constant w.

3.3.5 Canny edge detection

After updating the approximate pupil position $\hat{s}$ in the previous step, Canny edge detection [7] is performed in an area with dimensions $W_{AOI}xH_{AOI}$, centred around $\hat{s}$. A Gaussian filter is applied beforehand. The Canny edge detector identifies and locates points of sharp changes in pixel intensity, which characterise boundaries of objects in the image, and combines these points into thin line segments called edges. It transforms the AOI into a binary image, where all pixels that belong to an edge, also known as edge points, have been given a value of 1 and all other pixels a value of 0. An individual edge is defined as any collection of 8-connected edge points.

3.3.6 Morphological operation

In the next part of the algorithm, a number of processing steps are performed on the detected edges, which attempt to filter out any edge that does not belong to the pupil-iris boundary. In order to speed up and simplify some of these processes, all edges need to be thinned to the minimum amount of pixels required to define it. During Canny edge detection, non-maximum suppression will have already significantly sharpened the edges, but not yet sufficiently for our intentions. So two morphological operations are applied to the image, that trim the edges to single pixel thickness. Any edge points erased by this operation are not entirely discarded, but are given a special tag instead. Before fitting an ellipse on a sub-set of the detected edges, these removed edge points are restored in order to obtain a more accurate fit.

3.3.7 Edge selection

The first edge filter that is implemented is based on the fact that the pupil contour is likely to encircle or at least be close to the pupil position prediction $\hat{s}$. So algorithm selects a sub-set of edges that are located at an acceptable distance from the pupil position estimate by sending out rays from $\hat{s}$ in eight directions. The first edge that each ray encounters along its path is accepted, as well as any other edges that they come across after the first, provided they are within a radius:

$$r = \frac{\Delta L}{2} \tag{3.29}$$

By allowing the ray to continue beyond r when no edge was encountered, the method becomes more robust against pupil size predictions that are too small. Furthermore, by accepting all encountered edges within a radius r, and not just the first edge a ray runs into, we avoid detection failure when $\hat{s}$ happens to be partially enclosed by non-pupil edges. Furthermore, very small edges are ignored for edge selection. This threshold is determined by the edge window length parameter N_l.

3.3.8 Edge classifcation

In the final step of our pupil detection algorithm, an ellipse needs to fitted on a combination of the previously selected edges. In case multiple edges are available, it might be necessary to fit more than one ellipse and then choose the optimal one based on a few criteria. Given n edges, the maximum number of ellipses required to fit is equal to the total number of edge combinations:

$$\sum_{1 \leq k \leq n} \binom{N}{k} = 2^n - 1 \tag{3.30}$$

This value increases exponentially with respect to n, so it is important to keep the number of edges as small as possible. For this reason, a classification scheme is applied that divides edges into two categories: those that are part of the pupil-iris contour and those that are not. Edges that belong to the latter category are discarded. This classification is based on a number of different edge features.

Firstly, the length L of an edge, which is approximated by the sum of the distances between 8-connected neighbours. If the relative arrangement of two neighbouring edge points is in one of the four cardinal directions then the distance between them is 1, if it is in one of the four intercardinal directions then it is $\sqrt{2}$. This length measure is sufficiently accurate due to the morphological operations that have been applied, which have left the edges with minimum thickness.

Secondly, the variance of radius σ_r from each edge point to $\hat{s}$ tells us about the orientation of an edge. It will be close to zero if the edge encircles $\hat{s}$ at a distance that is roughly constant, i.e. the pupil boundary. Other edges are likely to have different orientations, leading to larger σ_r values.

The radial gradient G_r is mean gradient of edge points calculated in radial direction from $\hat{s}$ and is calculated at each edge point by looking at the direction vector between the edge point and the position estimate $\hat{s}$. The difference in intensity between two pixels that lie along this direction on opposite sides of the edge point is the measure for G_r at that particular position. The pixel value closer to $\hat{s}$ is subtracted from the one further away, so that a larger G_r is obtained for gradients that transition from dark to light the further

we move outwards. By considering the radial gradient, we add additional weight to edges belonging to the pupil boundary, because it marks the transition from dark to light in the radial direction, unlike many other edges.

Lastly, the intensity I is mean brightness of pixels on inside of edge curve. It is not simply given by the mean grayscale value of all edge points, but is instead calculated while taking into account the direction the edge curves in. Only pixels that lie on the inside of the edge curve are considered for I. Of course, for the pupil these pixels will be relatively dark, but for other edges this is not necessarily the case.

For each feature value F_i, a score is calculated through a function that is unique to each feature. We distinguish the two classes using a linear combination of all scores S_{tot}, where the classification is controlled by a constant threshold value. The total score value is computed by:

$$S_{tot} = \sum_{i}^{6} w_i g_i(F_i) \tag{3.31}$$

Where g_i is a Gaussian function with a certain set of parameters unique to feature i, and w_i is a weight that signifies the importance of that particular feature in classification. The Gaussian function has been chosen because it can easily be scaled in accordance with the frame-rate. The standard deviation of the function is increased with smaller frame-rates, because feature predictions are likely to be less accurate, causing feature values of pupil boundary edges to fall farther from zero. The distributions of feature values F_κ and F_{σ_r} have been limited to edges with $F_L \leq 0.75$, because the curvature and variance of radius are only relevant if the edge is reasonably long. Furthermore, when calculating the scores for κ and σ_r their weights are modified with F_L according to:

$$w^{'} = (1 - \beta F_L)w \tag{3.32}$$

Where β is another type of weight factor, whose value is determined alongside the other weights. Therefore, every weight is multiplied with the relevant certainty value.

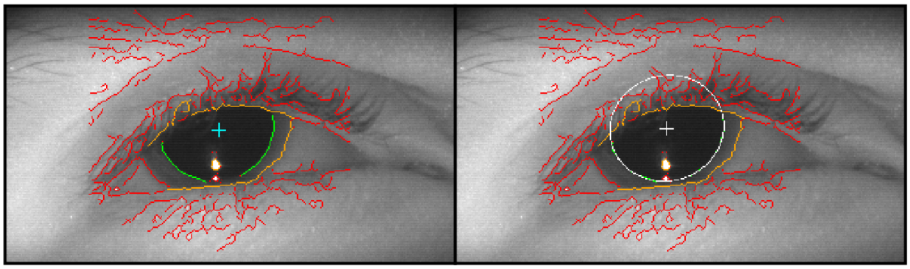

Figure 3.3: Edge classifcation example.

3.3.9 Edge segmentation

Owing to classifier, we are now able to distinguish between edges that make up the pupil outline and edges that do not. However, this leaves one important class of edges not dealt with, namely those that only partially belong to the pupil boundary. Before fitting an ellipse on these types of edges, we have to split them first at the transition point between pupil edge and some other feature (e.g. eyelid), otherwise the accuracy of the ellipse fit

will be adversely affected. For this purpose, three novel edge segmentation techniques have been developed, which are applied before edge classification is carried out.

Path segmentation

The first edge segmentation scheme is implemented based on the self-evident fact that the pupil is a non-intersecting closed shape. In other words, its perimeter is wholly described by a single non-branching path. So, if an edge consists of multiple branches, it definitely cannot exclusively be part of the pupil outline. Such edges are segmented by finding the non-branching path in the edge whose length comes closest to the circumference prediction $\hat{C}$ and separating it from the rest of the edge. To find this path, the edge is represented as an undirected graph, which refers to the mathematical structure consisting of a collection of vertices (or nodes) joined by edges with no orientation. The edge is converted to a graph through the following rules and definitions:

1. A branch vertex is any edge point that has three or more 8-neighbours
2. A terminal vertex is any edge point that has exactly one 8-neighbour
3. An arc is a collection of edge points that connects two vertices
4. Vertices that are 8-neighbours are combined into a single vertex

After creating the graph, recursive implementation of depth-first search is used to find every possible simple path, which is any path that does not traverse the same arc or vertex more than once, and then select the most optimal one. We locate all paths starting from the terminal vertices first and keep track of the vertices we already started with so we do not end with them. This prevents acyclic path repetitions, because for our purposes a specific path between two vertices is the same in the reverse direction.

Generally, several paths will be detected, one of which has to be accepted, based on a few criteria. Cyclic paths always take precedence over acyclic paths, since the ideal pupil outline is cyclic. However, if the pupil boundary is described by a cycle then it has to be wholly represented by that cycle, so the length L of such a cyclic path has to match the expected circumference of the pupil boundary, where the acceptable range is given by:

$$C_{min} \leq \hat{C}(1 - \delta_{\theta,C}) \leq L \leq \hat{C}(1 + \delta_{\theta,C}) \leq C_{max} \tag{3.33}$$

The lower limit of this criterion must not apply for acyclic paths, because multiple acyclic paths can make up the pupil boundary. Finally, any arcs that were not filtered out, but also not included in the final accepted path are not discarded. Instead, new graphs are made using these remaining edges and the process is repeated.

Curvature segmentation

Another property of the pupil-iris contour is its smooth curvature. Abrupt changes in curvature along its path are most likely caused by obstructions of the pupil periphery by the eyelid, eye lashes or corneal reflections. An algorithm exploits the curvature characteristic of the pupil boundary in order to separate it from these types of artefacts. The location of an occlusion or breakpoint in the pupil boundary is determined by checking if the curvature κ at a certain point on the perimeter is above an upper threshold (κ_{max}) or below a lower

threshold (κ_{min}). The pupil boundary is then segmented according to a number of heuristics that use the distances between detected breakpoints to detect different types of occlusions.

The method works by scanning the 8-connected environment of each edge point and assigning a vector label to the edge point depending on the relative position of its neighbour. After obtaining all direction labels, the curvature κ at each edge point is calculated. The curvature at a specific point on a curve is defined as the rate of change of the tangential angle ϕ with respect to the arc length s. This exact definition of κ is approximated by placing two windows on either side of the edge point we are calculating κ for and finding the mean unit vector for each window, where the size of a window is equal to the edge window length N_l. The difference in tangential angles $\Delta\phi$ between the two mean vectors, divided by N_l, is then the signed curvature at that point:

$$\kappa = \frac{d\phi}{ds} \approx \frac{\Delta\phi}{N_l} \tag{3.34}$$

The angle difference $\Delta\phi$ is then given by:

$$\Delta\phi = atan2(T_{2,y}, T_{2,x}) - atan2(T_{1,y}, T_{,x}) \text{with} \Delta\phi = \begin{cases} \Delta\phi - 2\pi & \text{if } \Delta\phi > \pi \\ \Delta\phi + 2\pi & \text{if } \Delta\phi < -\pi \end{cases} \tag{3.35}$$

For edges belonging to the pupil boundary, we expect that each edge point has the same curvature sign, since the boundary is elliptical. However, whether the majority sign is negative or positive depends on the scanning direction, which we do not control here. In order to properly set a lower curvature threshold it is crucial that each edge has the same majority curvature sign. For this reason, we count the number of positive and negative curvatures in the edge. If there are more negative curvatures than positive ones, all signs are inverted.

Now that we have a measure of the curvature for every edge point, we can use it to segment edges at the intersection between distinct features in the image. To do this, curvature segmentation algorithm works on the basis of only one simple rule: an edge is segmented at every break point. This makes it far less specific and requires fewer parameters. The only parameters we need to specify are the values for κ_{min} and κ_{max}. The circumference C of the pupil is inversely proportional to the mean of κ. The aspect ratio AR, on the other hand, is inversely proportional to the size of the range of κ. When the eye is looking straight ahead, the pupil can be reasonably approximated by a circle (i.e. AR = 1), which means that κ will be constant around the perimeter, so its range is minimal. However, when the eye is looking up, the pupil shape is more eccentric, so AR will be smaller. This causes the curvature at the antipodal points of the semi-minor axis to decrease and the curvature at the semi-major axis antipodes to increase, broadening the range of κ.

To ensure that κ_{min} and κ_{max} depend on the size and shape of the pupil, these parameters are turned into dynamic thresholds that are automatically updated according to predicted values $\hat{C}$ and $\hat{AR}$. The thresholds are based on the lower and upper limit of the κ range we would expect to get from an ellipse with a circumference and aspect ratio equal to $\hat{C}$ and $\hat{AR}$. In practice, these thresholds need to be offset (up and down) by a few degrees to account for inaccuracies in our predictions and for natural deviation of the pupil from the ellipse shape. The final thresholds are therefore determined by:

$$\kappa_{max} = h(\hat{C}(1 - \delta_{\theta,C}), \hat{AR} - \delta_{\theta,AR}) + offset \tag{3.36}$$

$$\kappa_{min} = h(\hat{C}(1 - \delta_{\theta,C}), \hat{AR} - \delta_{\theta,AR}) - offset \tag{3.37}$$

Where $h(C, AR)$ is the fitted function. The variable $\delta_{\theta,AR}$ is calculated by:

$$\delta_{\theta,AR} = (1 - c_{AR})(1.0 - \delta''_{\theta,AR}) + \delta''_{\theta,AR} \tag{3.38}$$

The pupil-iris outline is detected by Canny edge detection as one continuous edge that is also part of the upper eyelid. Through curvature segmentation we are able to split the edge up into two sections corresponding to the distinct features, after which the pupil edge can be successfully identified with edge classification.

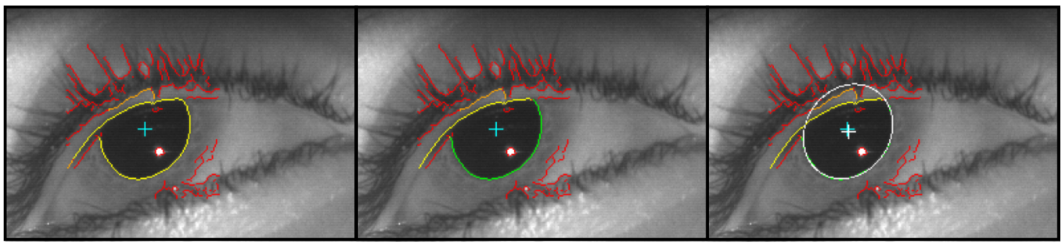

Figure 3.4: Curvature segmentation example.

Length segmentation

If the pupil circumference prediction in a given frame is accurate, then no edge that entirely resides on the pupil boundary should be longer than this prediction. When the edge length L is greater than $\hat{C}$, the edge is segmented so that one of the two parts has a length that is approximately equal to $\hat{C}$. The separation is graphically shown in $FIGURE - REF$, by temporarily dividing the edge into three sections and cutting the edge in such a way that the length of the edge body (II) plus the length of either of the two edge tails (I or III) is equal to $\hat{C}$, where $\hat{C} = 2\pi\hat{r}$ the figure.

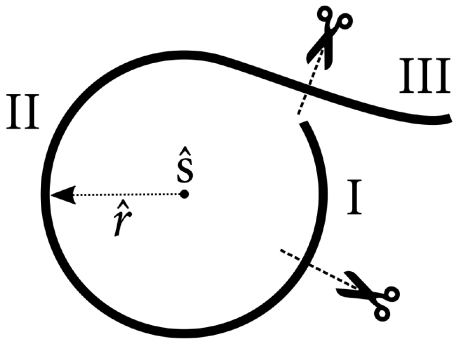

Figure 3.5: Length segmentation procedure.

3.3.10 Ellipse fitting

Having segmented and classified the edges obtained with Canny edge detection, we can now fit an ellipse on one or more of them, using one of several possible methods. Since numerous segmentation routines have been performed to remove any potential outliers, direct least squares fitting method can be applied[11]. Owing to its computational efficiency, we are

able to fit an ellipse multiple times in the same frame and then choose the most optimal one. Fixed maximum number of available edges to be classified as lying on the pupil-iris contour is set to 4 by default, which translates to 15 possible combinations, favouring edges with a higher score. Fitting that many ellipses in one frame is still too demanding, so their numbers will be further reduced by requiring that the expanse of each edge combination must reasonably correspond with width and height predictions of the pupil. This also means that that a significant portion of the pupil-iris contour is required to be visible in the image and has been detected before fitting an ellipse, because the fit will be gravely inaccurate otherwise. The range of the edge combination is calculated by finding the minimum and maximum x and y-positions in the collection of edge points and set the following criterion:

$$0.3(\hat{W} - \Delta l) \leq x_{max} - x_{min} \leq \hat{W} + \Delta l \tag{3.39}$$

$$0.3(\hat{H} - \Delta l) \leq y_{max} - y_{min} \leq \hat{H} + \Delta l \tag{3.40}$$

Where:

$$\Delta l = \frac{\hat{C}\delta_{\theta,C}}{\pi} \tag{3.41}$$

These thresholds ensure that cases where the vast majority of the pupil is obscured by the eyelid are ignored. A limit is also imposed on how many ellipses we are allowed to fit in one frame, which is fixed at 6 by default. If there are more possible fits available, then we choose the edge combinations with a combined length that is closest to $\hat{C}$.

The properties of every ellipse are subsequently calculated from the general equation of the ellipse using rotation transformation. This includes the position, semi-major and minor axes, rotation angle and bounding box dimensions. The circumference is computed by Ramanujan's second approximation. Immediately after obtaining these characteristics, each ellipse passes through a series of filters that will judge the size, shape and quality of the fit. The circumference of the ellipse has to fall within the C_{min} and C_{max} limits and its aspect ratio should be larger than AR_{min}. These bounds have been empirically determined from data set. A linear function is created that acts as a circumference threshold, which is given by:

$$C'_{max} = k(AR - 1) + C_{max} \tag{3.42}$$

Where k is assigned a value of 154. The circumference of the fit has to be less than the circumference limit calculated by this function. The criterion is established that δ_C and δ_{AR} are not allowed to respectively exceed $\delta_{\theta,C}$ and $\delta_{\theta,AR}$, otherwise the fit is rejected. Another filter examines the number of points that the ellipse was fitted on with respect to the fit's circumference. Therefore the following threshold is enforce for the edge length:

$$L \geq 0.3\hat{C}(1 - \delta_{\theta,\hat{C}}) \tag{3.43}$$

The last filter works on the basis of the error between the fit and fitted points. For ellipse fits on the pupil-iris edge, a small fit error for all fitted edge points is expected, since we are able to adequately approximate the pupil shape by an ellipse. So a large fit error for a particular edge point would indicate that the ellipse is not at all or not entirely fitted on the pupil boundary. The accuracy of the pupil fit can already significantly diminish if the edge combination contains just a few outliers, because of the high sensitivity of direct least

squares compared to other, slower, ellipse fitting methods. For this reason, only ellipse fits where all edge points in the set have a small fit error should be accepted. On the other hand, it is undesired to reject a fit because of just one outlier. As a compromise, the fit errors of the $0.05C$ largest outliers of the set are considered. If their average fit error is above a given threshold, the ellipse fit is rejected.

In many cases, only one ellipse fit remains at this stage which we then consider to represent the pupil. However, it also frequently occurs that there are still a few ellipse fits left to choose from. The final choice is made through a similar strategy employed during edge classification. Each ellipse is assigned a score based on a number of its features, after which the ellipse with the highest score is selected as the pupil representation.

3.4 Conclusion

For the purpose of the work, it was necessary to implement solution, which can reliably find precise centre of pupil. However, most commonly used algorithms described in this chapter are suitable only for general eye localisation and thus accuracy of these algorithms is insufficient for purposes of this thesis. Therefore, solution is proposed which will combine two algorithms into one. At first, Timm & Barth's algorithm will be used for eye localisation in each frame. Afterwards, region of interest will be cropped from the frame around the eye and Terence Brouns's algorithm will be applied in order to determine precise centre of the pupil. In conclusion, none of the previously described algorithms provides necessary accuracy, but combination described in this section should ensure required accuracy.

Chapter 4

Design and implementation

For the purpose of this thesis, there was necessity to implement solution, which will be able to localise absolute centre of pupil in human eye. This chapter describes the practical aspect of the thesis. Based on the proposal from chapter 3, the process of development and implementation details is specified in the following sub-chapters. The interconnection of individual parts of the program is outlined here. Additionally, this chapter describes its essential enhancements, program parameters and how to work with it.

4.1 Implementation details

Implementation was done in programming language C++11 using mainly OpenCV library, version 3.3.1 [25]. OpenCV is open source library for computer vision and was used for image handling, and feature extraction. Additional libraries used include Eigen 3.3.3 as a library for linear algebra: matrices, vectors, numerical solvers, and related algorithms, while libserialport takes care of the details related to usage of serial ports. Program was developed on operating system Ubuntu 17.10 with g++ compiler version 7.2.

4.2 System structure

Implemented system must be able to handle few minor tasks, other than the eye centre localisation itself. Once input from camera is available, it is necessary to wait for its stabilisation, in a sense that objects in frames stop moving. Afterwards, process of finding eye centres begins. The goal is to combine both algorithms described in chapter 3. At first, Timm & Barth's algorithm is used to determine approximate position of pupil in a frame. As mentioned before, this algorithm works quite fast and offers great results, however, it is unable to determine precise centre of pupil. Another requirement in order to achieve precise results, is removing specular highlight points in the frame, created from the platform's lightning. In a second step, Terence Brouns's algorithm is used in a specified region determined by Timm & Barth's algorithm as this algorithm only works on images containing solely eye itself. Accuracy of this algorithm improves with iterations, however, a limit must be present to assure realistic computational time. Once precise centre of pupil is identified, distance from frame centre is calculated and subsequently, based on these values, commands are generated and sent to platform, changing it's position to correct location. Simplified scheme of this process is available in figure 4.1.

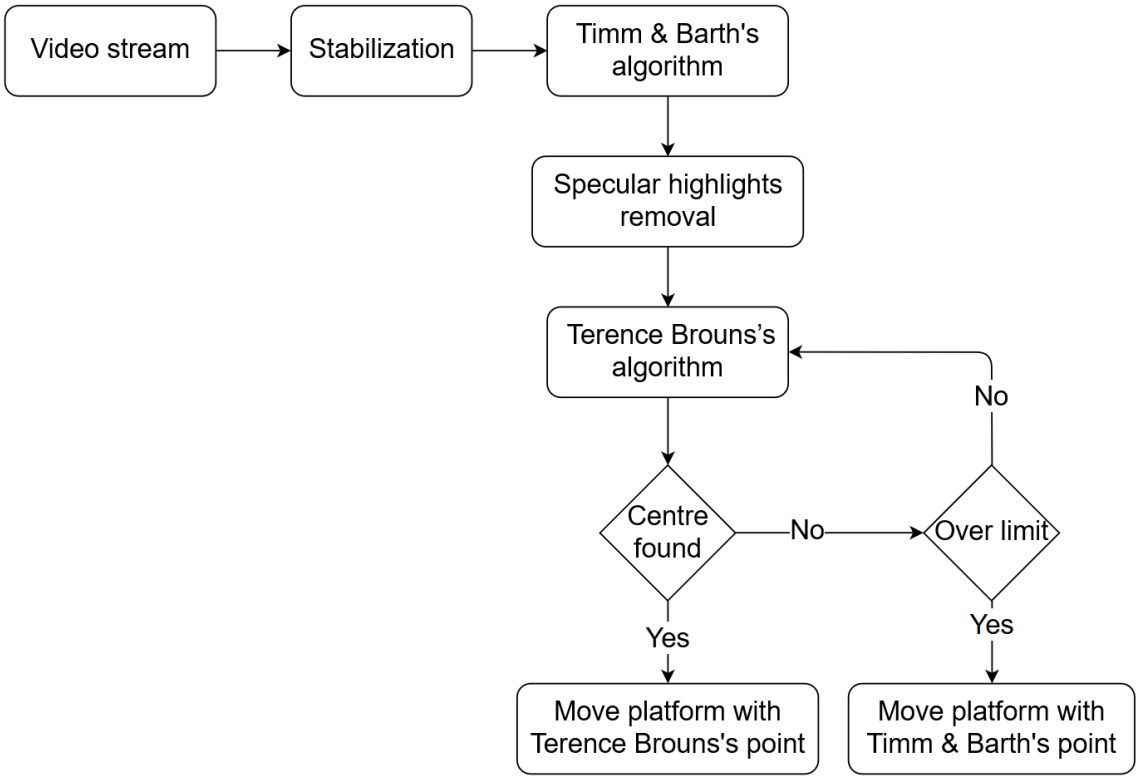

Figure 4.1: UML diagram of system scheme.

4.3 Program arguments

Implemented program can be launched with few optional parameters. If none are used, program will launch in default mode, using camera with index 0 and platform's port with name */dev/ttyUSB0*. In second mode, program can use image instead of camera input. In this case, it is also necessary to specify a path to input file. Lastly, it is possible to active debug mode and display intermediate data about localisation. This includes displaying results of individual frames from each algorithm that is used. In this mode, user can store current frame by pressing *f* button on his keyboard, or *c* button to exit debug mode. All available arguments can be seen in table 4.1.

Argument	**Meaning**
-h	Help mode, displaying program information
-c [index]	Set camera index mode
-p [port]	Port name for platform control
-i [path]	Image mode, use eye centre localisation on image located on given path
-d	Debug mode displaying intermediate data

Table 4.1: Table of program arguments values of parameters.

4.4 Camera capture

After the program launch, class `CameraCapture` encapsulates all of the functions related to the camera. An OpenCV's module `VideoCapture` is used to read input from the camera and frame by frame. Because of the possibility that device can have multiple cameras, index is used to determine which camera to use. This index is specified as one of the program arguments with default value being zero. Other important options are also set here, such as frame height, frame width and frames per second. Secondary goal of this class is also to start next step of the calculation, once input from the camera is stabilised. This is implemented via method `isStable()` where every 125 μs current frame is stored and then last three frames are compared. Mask is created from absolute difference between pixels of these frames. Once the number of pixels in this mask is below set threshold, the input is declared stable.

```
// Calculate per-element absolute differences between frames
absdiff(framesBuffer[0], framesBuffer[1], difference1);
absdiff(framesBuffer[1], framesBuffer[2], difference2);

// Calculate per-element bit-wise disjunction
bitwise_or(difference1, difference2, differenceTotal);

// Grayscale total difference
cvtColor(differenceTotal, differenceTotal, CV_BGR2GRAY);

// Find pixels above threshold and add erosion
threshold(differenceTotal, differenceTotal, THRESHOLD, 255, THRESH_BINARY);
erode(differenceTotal, differenceTotal, getStructuringElement(MORPH_RECT, Size(3, 3)
    ));

if (countNonZero(differenceTotal) < stabilityThreshold) {
    return true;
}
```

4.5 Specular highlights removal

In order to acquire precise iris and retina images, platform must contain of several strong lightning sources. These sources often leave reflections in captured frames. Even today, removal of specular highlights is still very open problem, especially in eye. Several complex approaches are available that solve this issue in a specific scenario. For example, authors Yang, Wang and Ahuja proposed real-time specular highlight removal using bilateral filtering. This method is based on an observation that the maximum fraction of the diffuse colour component in local patches in colour images changes smoothly. Using this property, values of the specular pixels are estimated by directly applying low-pass filter [41]. Many similar complex methods exist, however, they all have quite high computational demands, as they usually require several seconds to complete. Another issue is a possibility, that these methods are not suitable for every type of images and thus, they can potentially even degrade the input image. Therefore only a simple and reliable solution is proposed here, utilising powers of OpenCV library. Results of this method can be seen in figure 4.2, where an image on the left is raw input from the camera with highlighted regions containing specular highlights and on the right side is an image after application of highlight removal. While results are not perfect, it still considerably improved accuracy of eye centre localisation. [29]

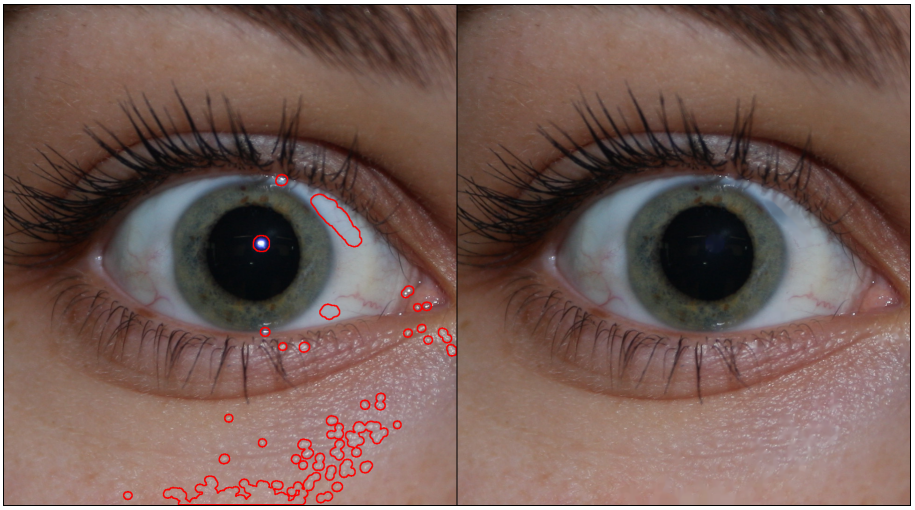

Figure 4.2: Application of specular highlight removal.

First of all, representation of colours in pixels is converted to HSL model, where individual pixels are represented through hue, saturation and lightness [16]. Points with lighting value close to maximum are typically strong sources of light. This threshold is calculated dynamically in a sense that only small percentage of brightest pixels are suitable. In the next step, noise is reduced using Gaussian blur. Then, mask is calculated with points above this threshold. This selection is then dilated through advanced morphological transformations, using an erosion and dilation as basic operations. Afterwards, restoration of the selected region in an image using the region neighbourhood with OpenCV method `inpaint()` is performed. Restoration is done using Navier-Stokes based method which uses series of equations to calculate missing pixels according to content of surrounding pixels. This process is implemented in class `SpecularHighlightsRemoval` and called via method `inpaintHighlights()` with current frame. [23]

```
// Remove noise
GaussianBlur(src, blur, Size(), 2, 2);

// Calculate the mask
threshold(brightness, mask, minBrightness, 255, THRESH_BINARY);

// Dialte a bit the selection
Morphology(mask, mask, MORPH_DILATE, dilateSize);

// Use OpenCV inpainting
float radius = 5.0;
inpaint(src, mask, dst, radius, INPAINT_NS);
```

4.6 Implementation of Timm & Barth's algorithm

Acquired frames from class `CameraCapture` are sent to class `Detector` via its method `detectAndDisplay()`. Input frame is here flipped and then, as described in section 4.2, process of locating eye centres starts with Timm & Barth's algorithm as described in section 4.6. Class `EyeLocalisation` encapsulates all of the functions related to its computation. Algorithm can be summed up to following points:

1. Gradient calculation
2. Calculation of significance for each point
3. Dynamic threshold calculation
4. Evaluation of all candidate points above threshold
5. Creation of a vector from each point to each candidate point
6. Calculation of the scalar component
7. Find the point with the best rating

At first, frame is converted to grey scale and re-sized to 100px width while keeping the aspect ratio. Subsequently, gradient is computed at first for original matrix as well as for transposed matrix. Authors use MatLab gradient function, which can be rewritten into C++ and OpenCV in method `computeGradient()` as following:

```
Mat out(in.rows, in.cols, CV_64F);
for (int i = 0; i < in.rows; i++) {
    const auto *row = in.ptr<uchar>(i);
    auto *outRow = out.ptr<double>(i);

    outRow[0] = row[1] - row[0];
    for (int j = 1; j < in.cols - 1; j++) {
        outRow[j] = (row[j + 1] - row[j - 1]) / 2.0;
    }
    outRow[in.cols - 1] = row[in.cols - 1] - row[in.cols - 2];
}
return out;
```

This step is followed by computation of all the magnitudes from calculated gradients, which is then used to calculate threshold to frame in method `computeDynamicThreshold()`. To do this, we calculate a mean and standard deviation of magnitudes:

```
Scalar stdDevMag, meanDev;
meanStdDev(mags, meanDev, stdDevMag);
double stdDev = stdDevMag[0] / sqrt(mags.rows * mags.cols);
return gradientThreshold * stdDev + meanDev[0];
```

Afterwards normalisation of gradients is performed, in such a way, that points where magnitude is lower than calculated threshold are set to zero thus becoming irrelevant and the rests of points are normalised by corresponding value of magnitude. This step is followed by smoothing the frame using a Gaussian filter with parameter value sigma of 0.005, as this step proved to improve the results.

In next step, we create output matrix with the same size as input frame and initiate it to zeroes. Then it evaluates every possible centre for each gradient point `(x, y)`, where value is non-zero. This is done by creating a vector from each candidate point to the gradient origin. Then this vector is normalised and dot product is calculated. Finally, this values is multiplied by weight and this results represents evaluation of each candidate point. Described process is implemented in method `testPossibleCenters()`:

```
for (int cX = 0; cX < out.rows; cX++) {
    double *outRow = out.ptr<double>(cX);
    const double *weightRow = weight.ptr<double>(cX);
```

```
        for (int cY = 0; cY < out.cols; cY++) {
            if (x == cX && y == cY) {
                continue;
            }

            // Calculate a vector from the possible centre to the gradient origin
            double dX = x - cX;
            double dY = y - cY;

            // Normalise this vector
            double magnitude = sqrt(pow(dX, 2) + pow(dY, 2));
            dX /= magnitude;
            dY /= magnitude;

            // Calculate dot product of vector and gradient
            double dotProduct = max(0.0, (dX * gX + dY * gY));

            // Square and multiply by the weight
            outRow[cX] = pow(dotProduct, 2) * weightRow[cX];
        }
    }
```

Dot products are negative if the vectors are pointing in opposite directions. The gradient function used creates vectors that always point towards the lighter region. Since the iris is darker than the sclera, the vectors of the iris edge always point out. This means that at the centre they will be facing in the same direction as the vector. Anything pointing in the opposite direction becomes irrelevant. To fix this problem, negative values must be turned to zero, so they have no effect on the results, using `max()` function with `0`. After this, we find a point with maximum value from `out` matrix, which is centre of the eye. Lastly, considering that input frame has been re-sized, found centre point must be re-scaled back to original frame while maintain its original position using method `unscalePoint()`.

4.6.1 Mask enhancement

During testing, one of main issues with this algorithm proved to be false detection of eye centres on the edges of camera input, especially in the corners. Sources of these anomalies consists of groups of hair, shadow alignments and backgrounds having similar characteristics as pupil and creating better scored candidate points. Therefore a mask is applied over calculated candidate points, shown in figure 4.3. A rectangle is created in the middle of matrix, occupying 60% of width and height, in which points keep their current value. Values of candidate points outside of this rectangle linearly decrease from rectangle to the edge of camera input, down to 30% of their previous values. This simple enhancement effectively eliminated this anomaly, while still maintaining the ability to detect eye centres outside of created rectangle.

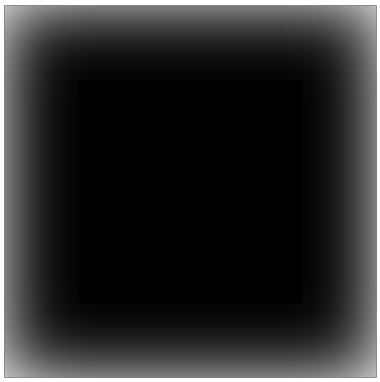

Figure 4.3: Applied enhancement mask.

4.7 Implementation of Terence Brouns's algorithm

After approximate eye centre has been located in section 4.6, it is necessary to find precise centre of pupil. To achieve this, Terence Brouns's algorithm described in section 3.3 is used with cut out region of the frame around the point found using Timm & Barth's algorithm. Reason for this is a fact that this algorithm only works with an area close to eye, as stated before, however it also has very high accuracy rate.

Application using Terence Brouns's algorithm is available as open-source. This application demonstrates algorithms capabilities and works as instrument for adjustment of parameters. On the other hand, main flaw of this application is lack of stability, as it tends to crash when launched with improper or intricate input images. From this application, files `eystalker.cpp` and `eyestaler.h` were extracted, as these files contain implementation of algorithm. However, for algorithm to operate properly, class `PupilCentralisation` was added, which encapsulates all required initiations and updates of parameters related to the implementation Terence Brouns's algorithm.

In basis, computation of pupil centre works on the basis of soft computing algorithms, in a sense that an initial set of parameters is generated and iteratively updated. Each new iteration is produced by stochastically updating estimations about certain pupil characteristics. As a result, parameters will gradually evolve to increase in precision, but higher number of iterations also means longer computation time. Furthermore, there is also a chance that parameters are kept updated in a loop and in this scenario pupil centre keeps jumping between few points without ever improving its accuracy. Because of this, implemented algorithm uses test of stabilisation after each iteration, in which result is declared final if its value has not significantly changed in last 4 iterations and algorithm ends. Therefore small deviation is allowed between iterations. In cases when no stable solution is found within total number of 16 iterations, last calculated solution is declared as final result. These number are optimal via testing.

Parameter	Value
gainAverages	0.005
gainAppearance	0.40
gainCertainty	0.25
gainPosition	0.75
cannyBlurLevel	4
cannyKernelSize	5
cannyThresholdLow/intensity	300.0
Radial cannyThresholdHigh	600.0
curvatureOffset	2.5
fitEdgeFraction	0.05
fitEdgeMaximum	4
thresholdFitError	0.60
glintWdth	12
thresholdCircumferenceMax	290
thresholdCircumferenceMin	60
thresholdAspectRatioMin	0.4
thresholdChangeCircumferenceUpper	0.12
thresholdChangeCircumferenceLower	0.03
thresholdChangeAspectRatioUpper	0.09
thresholdChangeAspectRatioLower	0.03
thresholdChangePositionUpper	6
thresholdChangePositionLower	3
thresholdScoreEdge	0.30
thresholdScoreFit	0.10
thresholdScoreDiffEdge	0.60
thresholdScoreDiffFit	0.10
windowLengthEdge	7
fitMaximum	6
cameraFrameRate	1

Table 4.2: Table of initial values of parameters.

Complete list of parameters parameters can be seen in table 4.2. Recommended values are used for initiation but at the same time some parameters are adapted for the platform's camera. Initial value of all parameters is set using `resetVariablesHard()`. To start computation, method `detect` is called with current frame. At first, current value of parameters is loaded using method `loadParameters()`. Based on current frame, parameters related to x axis are updated using method `updateAOIx()` and parameters related to y axis are updated using method `updateAOIy()`. After that, locating centre of pupil can start, by calling function `eyeStalker()` with current frame and parameters. This function returns precise position of pupil centre in a form of OpenCV *Point*. Once search is done, values of parameters are updated. Since this algorithm returns position of point $(0, 0)$ when centre is not found, it is necessary to check returned value and possibly call this function again with current frame. Further iterations are very fast, since parameters are already adjusted from first iteration. However, there also exists a possibility that algorithm is unable to locate

centre even after few iterations. In this scenario, system returns to the previous step and uses Timm & Barth's algorithm again.

4.8 Platform control

This section describes communication protocol between PC and position-able platform. First, method of calculating distance from pixels is presented, then platform protocol and its commands are described.

4.8.1 Calculating the distance

Calculating distance to move the platform is similar to triangles in the diagram 4.4, but for any accuracy, precise values of lens focal length and sensor size are required. Then the highlighted formula is used to compute the distance. The sensor size and focal length are both in millimetres, which is one ratio. The object distance and size are also both in millimetres, which is another ratio. Both are same equal ratio, thus making an equation 4.1.

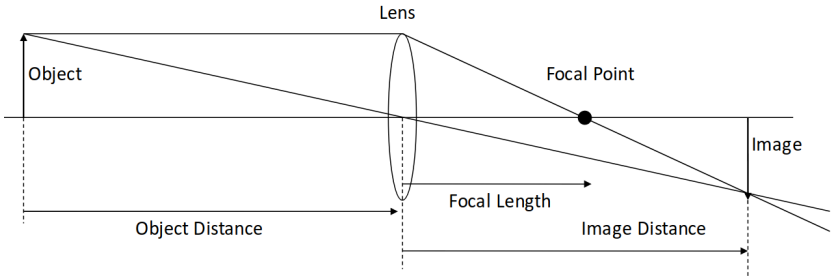

Figure 4.4: Lens distance calculation diagram.

$$\frac{Object\ height\ on\ sensor}{Focal\ length} = \frac{Real\ object\ height}{Distance\ to\ object} \tag{4.1}$$

By assuming expected position of a head, it is possible to calculate distance from the pixel difference of two points as seen in formula 4.1. Based on camera resolution and lens focal length in the platform, each pixel in both axes is converted to $112\mu m$ of necessary platform movement in that direction. This value is also contained in file `constants.h` and can be changed depending on the parameters of used camera. Several values are available for different cameras that were used during testing.

4.8.2 Details of movement commands

Platform uses a byte-oriented protocol. First byte determines the meaning of following byte. First byte is divided into two part. Higher part of byte (bits 4-7) determines a part or a peripheral of the motherboard that receives the message. Lower half of the byte is a command for the receiver. Commands start with literals $0x$ marking them as hexadecimal integers. For platform movement, two types of commands are available. At first, it is necessary to set rotation speed. Related commands of the platform are described in table 4.3 with following flags:

- f - force - set this value even of is over limits
- s - sign - set direction of rotation, 1 means left
- r - reserved
- v - value - 12-bit value of speed

bits	7.	6.	5.	4.	3.	2.	1.	0.
1. byte	f	s	r	r	v11	v10	v9	v8
2. byte	v7	v6	v5	v4	v3	v2	v1	v0

Table 4.3: Rotation speed commands.

For example, to set rotation to left, with speed of 128 degree/s, the following command is used:

$$0x \quad 40 \quad 80$$

Shift axis by
Afterwards, platform movement commands can be used. Related commands are described in table 4.4 with following flags:

- f - force - set this value even of is over limits
- s - sign - set direction of rotation, 1 means left
- r - reserved
- v - value - 20-bit value of shift in μm

bits	7.	6.	5.	4.	3.	2.	1.	0.
1. byte	f	s	r	r	v19	v18	v17	v16
2. byte	v15	v14	v13	v12	v11	v10	v9	v8
3. byte	v7	v6	v5	v4	v3	v2	v1	v0

Table 4.4: Shift axis commands.

For example, to shift to left, by 10 mm, the following command is used:

$$0x \quad 40 \quad 27 \quad 10$$

4.8.3 Platform functions

Class `PlatformControl` encapsulates all the functions related to the movement of the platform. Its method `controlPlatform()` is used to move platform. At the beginning, method `computeDistance()` is used to calculate distance to new location. This value is then converted to μm via method `pixelToMicroMeters()`, based on the distance in pixels from the camera. To communicate with serial port, C++ library *libserialport* is used. It is a minimal, cross-platform shared library written in C that is intended to take care of the OS-specific details when writing software that uses serial ports. It is licensed under the terms

of the GNU Lesser General Public License, version 3 or later. Port is open using 9600 baud rate, which is default for the platform. Then, following methods and their combination are used to move platform: `moveXLeft()`, `moveYLeft()`, `moveYUp()` and `moveYUp()` depending on relative position of camera centre. Each of these functions constructs a necessary command for the platform by setting up axis, direction and the distance to travel. If these commands are passed to the platform successfully, serial port is closed and freed.

Chapter 5

Testing & Evaluation

In the following chapter, the system implemented in the previous chapters will be evaluated. Most experiments were run simultaneously during implementation itself, however, only final results are presented. Thanks to this approach, it was possible to implement algorithms that achieved the best results.

5.1 Verification method

Implemented algorithm is developed for platform described in section 2.2, which expects frontal face images and it also has one specialised camera available for each eye. Therefore, the developed system expects approximately square images containing around half of the face with one eye. The following experiments used static images from databases, as well as real-time images from cameras. Requirements for successful test evaluation are following: Timm & Barth's algorithm is required to find any point in a pupil and Terence Brouns's algorithm is required to locate approximate centre of the pupil.

5.1.1 Data sets

Two databases were used for purpose of this work. Main decisive factors applied during selection of databases include resolution of images and requirement to contain face captured from the front. First of the databases is the SiblingsDB [8] containing different data sets depicting images of individuals related by sibling relationships. It contains a set of high quality images depicting 184 individuals. The images, with resolution 4256×2832 pixels, were shot by a professional photographer with uniform background and controlled lighting. The subjects are voluntary students and employees of the Politecnico di Torino and their siblings, in the age range between 13 and 50 (average 23.1). Second used database is intern database of FIT BUT called STRaDe EBD - Irises. It contains image of eyes and irises of 110 individuals and each of them has 6 images available. All of these are high-resolution with resolution 4752×3168 pixels. Hence, images from databases had to be manually modified accordingly to match expected input.

5.2 Database testing

The first part of testing is based on the use of databases. Main reasoning behind database usage is a fact, that it is more efficient for optimisation of the algorithm than using a

platform. In total, 68 images were selected and modified for testing, 41 from SiblingsDB and 27 from STRaDe EBD, with focus on choosing images with as much variety as possible. This number was due to complexity of modifying images as well as verifying results. Resolutions of these images is around 600×600 pixels for images from SiblingsDB and 2000×2000 pixels for images from STRaDe EBD. These images are available in appendix.

Figure 5.1: Results of implemented algorithm on images from SiblingsDB.

Due to complexity of finding eye centres, results were verified manually. Goal was to find approximate centre of pupil, which means locating precise pixel was not necessary. At the same time, results located near pupil edges were considered unsuccessful. Several examples of final results can be seen in figures 5.1 and 5.2. First column contains input images, second column are results of Tim & Barth's algorithm and last column are results of Terence Brouns's algorithm after application of specular highlights removal.

Figure 5.2: Results of implemented algorithm on images from STRaDe EBD.

5.3 Platform testing

The implemented system was also tested with the platform. Main objective of platform testing was ensuring that API for platform control works correctly. Second objective was to evaluate algorithm's accuracy. Since platform's camera is in the development at the moment, real-time video was acquired using various webcams. The main drawback of video footage compared to static shots is in quality of individual images where the faces in the video are potentially blurred and their resolution might be lower. In total, 12 subjects volunteered for testing, each scanned multiple times and with different webcams. Reason for this is a fact, that various parameters of these cameras would enable better evaluation and comparison of the algorithm's performance for each camera.

First tested camera is SoftKinetic DepthSense 325. However, even though it has 720p resolution, camera proved to be unable to focus on objects that were placed in close proximity. During platform testing, subjects placed their heads approximately 10 centimetres from the camera, which resulted in blurred images. Developed algorithm was unable to locate eye centre in every frame captured by this device, and therefore this camera proved to be unusable for testing.

Second tested camera is TRUST Exis Webcam with 480p resolution. This camera offers manual focusing and, despite its resolution, offered much better results than the previously used camera. Timm & Barth's algorithm was able to find eye centres in 87.5% instances of captured frames. On the other hand, Terence Brouns's algorithm failed in each of these instances, since resolution of acquired frames with this camera is insufficient. In conclusion, eye centralisation algorithm proved to be usable with this camera in the most cases. Results can be seen in figure 5.3, where first column contains raw images with axes highlighted, which is centre of the image. Second column contains results of Timm & Barth's algorithm and last column contain axes after corresponding platform movement.

Figure 5.3: Testing results for TRUST Exis Webcam.

Another tested camera is Genius WideCam F100 1080p. This device offers high resolution and 120° wide-angle optics. However, this wide-angle optics proved to skew the image and ultimately, worsen the results. In summary, Timm & Barth's algorithm could find correct eye centres only in 53.8% instances of captured frames. At the same time, Terence Brouns's algorithm was successful in 85.7% of these instances, since resolution of acquired frames is insufficient with this camera. Results can be seen in figure 5.4, where first column contains raw images with axes highlighted, second and third column contains results of Timm & Barth's and Terence Brouns's algorithms and last column contains axes after corresponding platform movement.

Figure 5.4: Testing results for Genius WideCam F100.

Last tested camera is Logitech Zeiss Tessar HD 1080p. This device provides highest quality of captured images from all tested cameras. It also offers automatic focusing feature, which can take up to 2 seconds. Therefore, all subsequent steps of the eye centralisation algorithm are a bit delayed. In conclusion, Timm & Barth's algorithm could find correct eye centres in 97.2% instances of captured frames. Similarly, Terence Brouns's algorithm was successful in 91.7% of captured frames. Results can be seen in figure 5.5, where first column contains images with axes highlighted, second and third column contains results of Timm & Barth's and Terence Brouns's algorithms. Last column contains axes after corresponding platform movement.

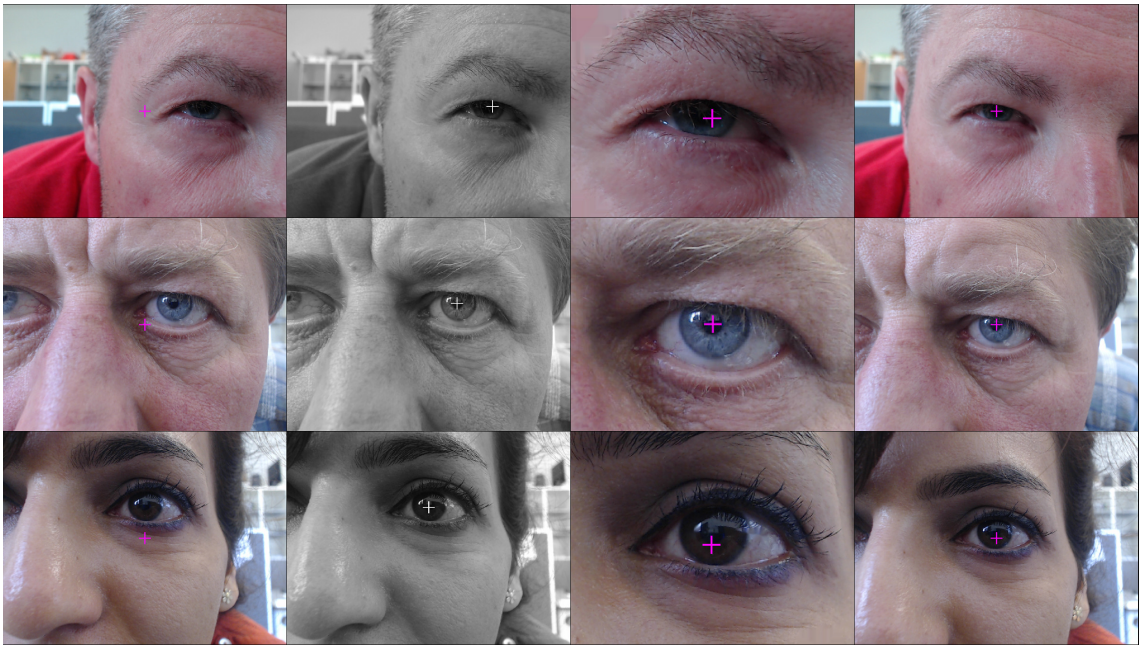

Figure 5.5: Testing results for Logitech Zeiss Tessar.

In summary, API and platform movement worked flawlessly throughout the testing. On the side note, in the current version, the platform is unable to detect its current position and therefore it is recommended to reset the platform's position, by moving to the centre of each axes. Otherwise, the platform is likely to hit the edge. Overall, accuracy of developed

algorithm for eye centre localisation greatly depends on the quality of the used camera. Most importantly, Logitech Zeiss Tessar had very high quality of captured frames and achieved nearly identical results to database testing, both in accuracy and time complexity.

5.4 Time complexity

Using images from databases, the implemented system can process 1 frame in 1189 ms on average. Stated results were achieved on a processor machine Intel Core i7 6700HQ and 8 GB RAM. However, since implemented program does not use external GPU or extensive parallelism with exception of library level parallelism, it is safe to assume that similar performance will be achieved on embedded device in the platform. Table 5.1 displays additional details about time complexity of algorithms.

Algorithm	**Minimum** $[ms]$	**Maximum** $[ms]$	**Average** $[ms]$
Tim & Barth's	429	631	541
Specular highlights removal	32	171	87
Terence Brouns	207	822	453
Complete system	789	1643	1189

Table 5.1: Time complexity of implemented system.

Similar results were achieved when cameras were used instead of static images. Required time for image processing was a bit lower, more precisely 977 ms on average. This is because algorithm uses preset scaling of images and therefore is invariant towards resolution of input images. Simpler images with easier eye centre localisation performed better since these images require less iterations of Terence Brouns's algorithm. On the other hand more complex or disarranged images tended to prolong computation time by few tens or hundreds milliseconds. For obvious reasons specular highlights removal also has significant dependency on quality of input images, but its time complexity is considerably lower. At the same time, Timm & Barth's algorithm was almost unaffected by this.

5.5 Evaluation of functionality

Overall results were very positive. As seen in figures 5.1 and 5.2, Tim & Barth's can reliably find pupil position, but not it's precise centre. However, it successfully found pupil in 100% of tested images. Secondly, Terence Brouns's algorithm can find precise centre of the pupil, but at the same time there are cases when it is unable to do so. This occurred in 10.3% of tested images and can also be seen in fourth row of figure 5.1. Main cause of this was severe reduction of resolution of images during their adaptation to suit platform's purposes. The majority of these images originated from SiblingsDB and their resolution was around 600×600 pixels, which proved to be often insufficient for Terence Brouns's algorithm. Better results were achieved with images from STRaDe EBD, since these images are available in better resolution, and they scored 92.6% with Terence Brouns's algorithm with being unable to locate eye centre in only 2 images. Based on this results, few final modification were implemented to the system. First, if Terence Brouns's algorithm is unable to find eye centre, previous value from Tim & Barth's algorithm is used as a result. Secondly, based on the described results, a limit was added to difference in distance between points from both algorithms and if this limit is broken, point from Tim & Barth's algorithm is used as

a result once again. Results of platform testing depended on the used camera. As could be seen in figure 5.5, when Logitech Zeiss Tessar was used, Timm & Barth's algorithm could find correct eye centres in 97.2% of cases and Terence Brouns's algorithm was successful in 91.7% cases. All other cameras performed worse. Since the platform will have even better camera, this does not pose a problem. Furthermore, computation time of entire process of eye centre localisation ranges from 800 ms to 1600 ms depending on difficulty of finding eye centres. Since user's head will be placed stationary for several seconds in front of the platform, achieved performance is satisfactory. To sum it up, this indicates that algorithm is usable for localisation of pupil centres.

5.6 Further work

Performed experiments have shown that even though implemented system achieved great results, there is also a room for improvements. Considering the fact that system will be used in real-time with patients, time complexity should be as low as possible. Even though 1189 ms is a decent result, better optimisation could further lower this number and thus making the experience more pleasant to patients. In addition, accuracy could also be improved. Input frames often contain reflections in pupil and surrounding areas which can considerably lower accuracy and are much harder to remove than specular highlights. A significant number of algorithms exist that can remove such reflections, but this operation remains non-trivial task. Furthermore, these algorithms are usually application specific and thus reliable implementation for eye could be quite difficult. Lastly, further testing and adjustments of parameters could also improve accuracy, especially for Terence Brouns's algorithm. For practical usage, it might also be necessary to adjust some parameters to camera on platform, as described in section 4.

Chapter 6

Conclusion

This work dealt with eye centre localisation. Features and characteristics of eye, commonly used scanning methods and scanning devices are described in chapter 2, as well as the device under development for acquirement and recognition of eye iris and eye retina. General approaches for eye localisation with basic technology are described in chapter 3. Chapter 4 focuses on implementation of selected methods. Implemented system is evaluated on data sets and tested with platform in chapter 5. Achieved results are then analysed.

This work showed that there are many methods for eye localisation and they work quite accurately and reliably. Implemented algorithm made by Timm & Barth does this as well and has relatively low performance requirements. However, none of these methods is designed for localisation of absolute centre of pupil. Therefore, combination of multiple methods is required and objective of second method is to determine absolute centre of pupil, once approximate region of pupil is located. The method that was chosen is Terence Brouns's algorithm, which does not work with unsuitable images and therefore works great as subsidiary algorithm. This combination achieved very positive results, only failing on few disarranged and complex images.

To sum it up, implemented solution shows that a eye centring and positioning a platform can be solved with single-board computer along with a camera. C++ and the OpenCV image manipulation library were used for its implementation. The overall results from testing show, that implemented system can be used, with considerable success, for controlling position-able platform for eye centring. Further improvement of data sets and some adjustments of parameters could lead to real world usage.

Bibliography

[1] Confocal Digital Ophthalmoscope F-10. [Online; visited 25.4.2018]. Retrieved from: http://usa.nidek.com/products/scanning-laser-ophthalmoscope/

[2] Non-Mydriatic Auto Fundus Camera AFC-330. [Online; visited 25.4.2018]. Retrieved from: http://www.nidek-intl.com/product/ophthaloptom/diagnostic/dia_retina/afc-330.html

[3] Albert, D.; Miller, J.: *Albert & Jakobiec's Principles & Practice of Ophthalmology*. Saunders. 2008. ISBN 978-1416000167.

[4] Asadifard, M.; Shanbezadeh, J.: Automatic Adaptive Center of Pupil Detection Using Face Detection and CDF Analysis . [Online; visited 31.12.2017]. Retrieved from: http://www.iaeng.org/publication/IMECS2010/IMECS2010_pp130-133.pdf

[5] Bowling, B.; FRCSEd(Ophth); FRCOphth; et al.: *Kanski's Clinical Ophthalmology: A Systematic Approach*. Saunders Ltd.. 2015. ISBN 978-0702055720.

[6] Brouns, T.: Robust Video-Based Eye Tracking Using Recursive Estimation of Pupil Characteristics. [Online; visited 7.2.2018]. Retrieved from: https://pdfs.semanticscholar.org/32e3/f4daecf5c0d3d4a33d8f9acfd34e2c043bca.pdf

[7] Canny, J.: *A Computational Approach to Edge Detection*. 1986. pAMI-8(6):679–698.

[8] CG&VC: Siblings Database. [Online; visited 15.4.2018]. Retrieved from: https://areeweb.polito.it/ricerca/cgvg/siblingsDB.html

[9] Commons, W.: Ophthalmoscope and otoscope combination. [Online; visited 14.5.2018]. Retrieved from: https://commons.wikimedia.org/wiki/File:Ophthalmoscope_Otoscope08.JPG

[10] Commons, W.: Schematic diagram of the human eye en-edit. [Online; visited 31.12.2017]. Retrieved from: https://commons.wikimedia.org/wiki/File:Schematic_diagram_of_the_human_eye_en-edit.png

[11] Csaba L. Martonyi, R. F. M., Charles F. Bahn: *Slit Lamp: Examination and Photography*. Time One Ink, Ltd.. ISBN 978-0615165196.

[12] Csillag, A.: *Atlas of the sensory organs : functional and clinical anatomy*. Totowa, N.J: Humana Press. 2005. 978-1-58829-412-8.

[13] DAS; Ravindra: *Biometric technology: authentication, biocryptoraphy, and cloud-based architecture*. FL : CRC press. 2015. ISBN 978-1-4665-9245-2.

[14] Drahanský, M.; R, K.; T., M.: Elektronické zařízení pro snímání obrazu sítnice a duhovky oka. 2015. [Online; visited 29.3.2018].
Retrieved from: http://www.prolekare.cz/pdf?id=57173

[15] Encyclopedia, W. T. F.: Fundus photography. [Online; visited 24.4.2018].
Retrieved from:
https://en.wikipedia.org/wiki/Fundus_photography#/media/File:Fundus_photograph_of_normal_right_eye.jpg

[16] Encyclopedia, W. T. F.: HSL and HSV. [Online; visited 31.3.2018].
Retrieved from: https://en.wikipedia.org/wiki/HSL_and_HSV

[17] Encyclopedia, W. T. F.: Slit lamp. [Online; visited 24.4.2018].
Retrieved from: https://en.wikipedia.org/wiki/Slit_lamp#/media/File:Retina_Group_slit_lamp_(side_view).jpg

[18] Erbaugh, J. K.: *Principles of ophthalmoscopy*. C.C. Thomas. 1958.

[19] Hamouz, M.; Kittler, J.; Kamarainen, J.; et al.: Feature-based affine-invariant localization of faces. 2005. [Online; visited 31.12.2017].
Retrieved from:
http://ieeexplore.ieee.org.ezproxy.lib.vutbr.cz/document/1471713/

[20] Hansen, D.; Qiang Ji, D.: In the Eye of the Beholder: A Survey of Models for Eyes and Gaze. [Online; visited 31.12.2017].
Retrieved from:
http://ieeexplore.ieee.org.ezproxy.lib.vutbr.cz/document/4770110/

[21] Hansen, D. W.; Ji, Q.: In the Eye of the Beholder: A Survey of Models for Eyes and Gaze. 2009. [Online; visited 15.4.2018].
Retrieved from: https://ieeexplore.ieee.org/document/4770110/

[22] Huang, J.; Wechsler, H.: Visual routines for eye location using learning and evolution. 2000. [Online; visited 31.12.2017].
Retrieved from:
http://ieeexplore.ieee.org.ezproxy.lib.vutbr.cz/document/843496/

[23] Kaehler, A.; Bradski, G.: *Learning OpenCV 3: Computer Vision in C++ with the OpenCV Library*. O'Reilly Media. 2016. ISBN 978-1491937990.

[24] Moriyama, T.; Kanade, T.; Xiao, J.; et al.: Meticulously detailed eye region model and its application to analysis of facial images. 2006. [Online; visited 31.12.2017].
Retrieved from:
http://ieeexplore.ieee.org.ezproxy.lib.vutbr.cz/document/1608037/

[25] OpenCV-team: OpenCV 3.3.1 Documentation. [Online; visited 1.1.2018].
Retrieved from: https://docs.opencv.org/3.3.1/

[26] Patrick J. Saine, M. E. T.: *Ophthalmic Photography: Retinal Photography, Angiography, and Electronic Imaging*. Butterworth-Heinemann Medical. 2001. ISBN 978-0750673723.

[27] Reale, M. J.; Canavan, S.; Yin, L.: A Multi-Gesture Interaction System Using a 3-D Iris Disk Model for Gaze Estimation and an Active Appearance Model for 3-D Hand Pointing. 2011. [Online; visited 31.12.2017].
Retrieved from:
http://ieeexplore.ieee.org.ezproxy.lib.vutbr.cz/document/5720546/

[28] Remington, L. A.: *Clinical Anatomy and Physiology of the Visual System*. Butterworth-Heinemann. 2012. ISBN 978-1437719260.

[29] Russ, J. C.; Neal, F. B.: *The Image Processing Handbook*. CRC Press. 2016. ISBN 9781138747494.

[30] Ryan, S.; Schachat, A.; Wilkinson, C.; et al.: *Retina*. Saunders. 2013. ISBN 978-1455707379.

[31] Scanlon, P. H.; Sallam, A.; van Wijngaarden, P.: *A Colour Atlas of Optic Disc Abnormalities*. Wiley-Blackwell. 2017. ISBN 978-1119058953.

[32] Scanlon, P. H.; Sallam, A.; van Wijngaarden, P.: *A Practical Manual of Diabetic Retinopathy Management*. Wiley-Blackwell. 2017. ISBN 978-1119058953.

[33] Timm, F.; Barth, E.: Accurate eye centre localisation by means of gradients. [Online; visited 31.12.2017].
Retrieved from:
http://www.inb.uni-luebeck.de/publikationen/pdfs/TiBa11b.pdf

[34] Valenti, R.; Gevers, T.: Accurate eye center location and tracking using isophote curvature. 2008. [Online; visited 31.12.2017].
Retrieved from:
http://ieeexplore.ieee.org.ezproxy.lib.vutbr.cz/document/4587529/

[35] Valenti, T., R. ; Gevers: Accurate Eye Center Location through Invariant Isocentric Patterns. 2012. [Online; visited 31.12.2017].
Retrieved from:
http://ieeexplore.ieee.org.ezproxy.lib.vutbr.cz/document/6109281/

[36] Wang, J.; Yin, L.; Moore, J.: Using geometric properties of topographic manifold to detect and track eyes for human-computer interaction. 2007. [Online; visited 31.12.2017].
Retrieved from:
https://dl-acm-org.ezproxy.lib.vutbr.cz/citation.cfm?doid=1314303.1314306

[37] Wang, J.-G.: Estimating the eye gaze from one eye. 2005. [Online; visited 31.12.2017].
Retrieved from: http://www.sciencedirect.com.ezproxy.lib.vutbr.cz/science/article/pii/S1077314204001134

[38] Wang, P.; Green, M.; Ji, Q.: Automatic Eye Detection and Its Validation. 2005. [Online; visited 31.12.2017].

Retrieved from:
http://ieeexplore.ieee.org.ezproxy.lib.vutbr.cz/document/4587529/

[39] Wanga, S.; Liu, Z.; Shen, P.; et al.: Eye localization from thermal infrared images. [Online; visited 31.12.2017].
Retrieved from: http://www.sciencedirect.com.ezproxy.lib.vutbr.cz/science/article/pii/S003132031300112X

[40] Webb, R. H.; Hughes, G. W.; Delori, F. C.: *Confocal scanning laser ophthalmoscope*. vol. 26. OSA. 1987.

[41] Yang, Q.; Wang, S.; Ahuja, N.: Real-time Specular Highlight Removal Using Bilateral Filtering. 2010. [Online; visited 31.3.2018].
Retrieved from:
http://vision.ai.illinois.edu/publications/eccv-10-qingxiong-yang.pdf

[42] Zhonglong, Z.; Jie, Y.; Limin, Y.: A robust method for eye features extraction on color image. [Online; visited 31.12.2017].
Retrieved from: http://www.sciencedirect.com.ezproxy.lib.vutbr.cz/science/article/pii/S0167865505001170

Appendix A

CD Content

- doc/: Program documentation,
- samples/: Dataset,
- src/: C++ source code,
- testing/: Video examples of platform control,
- tex/: LaTeX source of this document,
- xmagdo01-dip.pdf: PDF file of this document.

Appendix B

Program usage

C++11 or newer compiler must be installed along with libraries:

- cmake 3.9,
- OpenCV 3.3.1,
- Eigen 3.3.4,
- libserialport.

To compile the source codes, run the following commands: `cmake . make`
Run with no arguments to print simple help:
`./eyeCentralisation`

```
Eye centralisation
Author: Patrik Magdolen, xmagdo01
Parameters:
    -c [index] - Set camera index mode
    -p [port] - Set port name
    -i [path] - Image mode, set input imagel
    -d - Debug mode displaying windows with partial results
```

Run with selected image in debug mode:
`./eyeCentralisation -i 1125_IRI_1_LX.jpg -d`

```
EYE: 2328 1568
PUPIL: 2387 1560
DURATION: 1130.04 [ms]
Moving y axis up by 10000 um
Command that will be passed = 0x402710
```

Run using camera with index 1 in debug mode:
`./eyeCentralisation -c 1 -d`

```
EYE: 1152 980
PUPIL: 1158 968
DURATION: 977.11 [ms]
Moving y axis up by 6580 um
Command that will be passed = 0x400103
```